

3

Les grammaires et les langages algébriques.

1 – Définitions de base.

On peut dire, de façon imagée, qu'une *grammaire est une machine à réécrire* c'est-à-dire, une machine à faire des substitutions, dans un sens très général qui appelle quelques précisions.

Les grammaires en général.

Une grammaire est un triplet $G = (\mathcal{V}, \mathcal{A}, R)$ où :


- \mathcal{V} est un alphabet **fini** dont les éléments sont appelés symboles non-terminaux ou **variables**;
- \mathcal{A} est un alphabet **fini**, disjoint de \mathcal{V} , dont les éléments sont appelés symboles terminaux ou **constantes**;
- R est un ensemble **fini** de couples $(\lambda, \alpha) \in (\mathcal{A} + \mathcal{V})^* \times (\mathcal{A} + \mathcal{V})^*$ appelés **règles** de production ou de réécriture, de G .
Une règle (λ, α) sera représentée de façon plus suggestive par $\lambda \xrightarrow{G} \alpha$, ou simplement $\lambda \longrightarrow \alpha$, lorsqu'il n'y a pas d'ambiguïté sur la grammaire en question.

Fonctionnement d'une grammaire.

- Une règle $\lambda \xrightarrow{G} \alpha$ agit sur un mot de la forme $\beta_1 \lambda \beta_2$ en remplaçant λ par α , ce que l'on schématise par

$$\beta_1 \lambda \beta_2 \xrightarrow{G} \beta_1 \alpha \beta_2.$$

- Une réécriture, que l'on appellera plutôt *une dérivation* (on dit encore *inférence*) est un enchaînement fini de telles opérations.
- L'ensemble des mots de \mathcal{A}^* que l'on peut obtenir par réécriture à partir d'un mot donné μ est *le langage engendré par G à partir de μ* .

 Il est naturel de considérer des grammaires de types particuliers, par la spécification de propriétés que doivent vérifier leurs règles : on peut obtenir ainsi une classification des grammaires et des langages engendrés. La classification la plus fondamentale est celle de Noam Chomsky, dont nous n'évoquerons que deux niveaux ici : les grammaires linéaires (à droite ou à gauche), qui engendrent les langages réguliers (cf. chapitre 2 et section 1.3 ci-dessous), et les grammaires "indépendantes du contexte" ou "algébriques", qui constituent l'objet de ce chapitre.

1.1 – Grammaires indépendantes du contexte.

Nous nous limitons aux grammaires dites *indépendantes du contexte* (context-free grammars) ou *algébriques* : leur forme est simple et leurs applications nombreuses, spécialement dans la définition des langages de programmation et donc la compilation des programmes.

Grammaires indépendantes du contexte ou algébriques

Une grammaire est dite *indépendante du contexte* ou *algébrique* lorsque le premier membre de ses règles est réduit à une variable, c'est-à-dire lorsqu'elles sont de la forme $X \xrightarrow[G]{} \alpha$ avec $X \in \mathcal{V}$ et $\alpha \in (\mathcal{A} + \mathcal{V})^*$.

Nous ne considérerons que des grammaires indépendantes du contexte et nous les appellerons simplement **grammaires**.

La définition présente les règles de façon individuelle. C'est sous cette forme que les règles interviennent le plus souvent dans la pratique, cependant il est commode de regrouper les règles relatives à une même variable.

Règles globales

La règle globale $X \xrightarrow[G]{} \mathbf{I}(X)$ de X dans G est définie par

$$\alpha \in \mathbf{I}(X) \text{ ssi } X \longrightarrow \alpha \text{ est une règle de } G$$

pour tout $\alpha \in (\mathcal{A} + \mathcal{V})^*$.

Exemple 1.

Soit $G = (\mathcal{V}, a + b, R)$ où $\mathcal{V} = S$. On peut décrire R

- ou bien comme un ensemble de règles individuelles, par exemple :

$$S \longrightarrow SbS$$

$$S \longrightarrow \varepsilon$$

$$S \longrightarrow a$$

$$S \longrightarrow aa$$

- ou bien par la règle globale correspondante :

$$S \longrightarrow SbS + \varepsilon + a + aa$$

Commentaires.

- $\mathbf{I}(X)$ est donc l'ensemble de tous les seconds membres des règles pour X : c'est un langage **fini**, puisque R l'est par définition. On peut donc décrire directement une grammaire comme un triplet $G = (\mathcal{V}, \mathcal{A}, \mathbf{I})$ où $\mathbf{I} : \mathcal{V} \rightarrow \mathcal{P}((\mathcal{A} + \mathcal{V})^*)$ est une substitution finie : cette possibilité ne se généralise pas très bien aux grammaires quelconques.

Bien que ce cas soit en général sans aucun intérêt pratique, rien n'empêche dans la définition que l'on puisse avoir $\mathbf{I}(X) = \emptyset$ pour certaines $X \in \mathcal{V}$: l'élimination de ces variables "vides" ne pose aucun problème . . .

1.2 – Dérivations dans une grammaire.

Les dérivations dans une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ sont les réécritures successives que l'on peut opérer sur les éléments de $(\mathcal{A} + \mathcal{V})^*$ par application de ses règles.

- Une *dérivation élémentaire* est la modification d'un mot par l'application d'une règle de la grammaire. Une dérivation élémentaire se définit pour $\beta_1 \in (\mathcal{A} + \mathcal{V})^*$, $\beta_2 \in (\mathcal{A} + \mathcal{V})^*$, $X \in \mathcal{V}$ par

$$\beta_1 X \beta_2 \xrightarrow[G]{1} \beta_1 \alpha \beta_2$$

pour toute règle $X \xrightarrow[G]{} \alpha$.

Ceci constitue donc seulement la *mise de la règle ρ dans le contexte (β_1, β_2)* : la modification du mot initial ne concerne qu'une de ses variables et n'impose aucune condition au contexte, c'est pourquoi une grammaire du type que nous étudions est dite "indépendante du contexte".

• Une dérivation de longueur i , notée $\beta \xrightarrow[G]{i} \gamma$ est une suite de i dérivations élémentaires qui s'enchaînent, qui se définit par récurrence sur i :

$$\begin{aligned} 0 : \beta &\xrightarrow[G]{0} \beta && \text{(pas d'application de règle)} \\ i \mapsto i + 1 : \beta &\xrightarrow[G]{i} \delta \xrightarrow[G]{1} \gamma && \text{(une application de règle de plus)} \end{aligned}$$

- On notera $\beta \xrightarrow[G]{*} \gamma$ pour signifier "il existe une dérivation $\beta \xrightarrow[G]{i} \gamma$ ".
- Lorsqu'il n'y a pas d'ambiguïté à craindre, on peut, comme il a déjà été dit, omettre la mention de G dans la désignation des règles et des dérivations : ceci donne des notations simplifiées $X \rightarrow \alpha, \beta \xrightarrow{i} \gamma$ et $\beta \xrightarrow{*} \gamma$. Nous utilisons cette écriture simplifiée dès maintenant.

Remarques et notations.

• Pour déterminer complètement une dérivation élémentaire, il faut préciser l'occurrence de la variable à laquelle on applique la règle. Nous le ferons en soulignant l'occurrence en question : $\beta_1 \underline{X} \beta_2 \xrightarrow{1} \gamma$ désigne alors une dérivation élémentaire lorsque $\gamma = \beta_1 \alpha \beta_2$ et $X \rightarrow \alpha$ est une règle de G .

• Une dérivation de longueur i fait exactement i applications de règles. Une dérivation de longueur 1 est une dérivation élémentaire.

Une dérivation de longueur 0, qui n'applique donc aucune règle, $\beta \xrightarrow{0} \beta$, s'appelle une *dérivation triviale*. Il y a une dérivation triviale $\beta \xrightarrow{0} \beta$ pour tout $\beta \in (\mathcal{A} + \mathcal{V})^*$ et il n'y en a pas d'autre si $\beta \in \mathcal{A}^*$. Lorsque l'on construit une dérivation $\beta \xrightarrow{i} \gamma$, la dérivation triviale initiale est simplement l'"installation" du mot β .

• Une règle agit par la substitution d'un mot à une variable; seules les variables sont capables de jouer un rôle actif dans une dérivation.

• Les dérivations se composent en enchaînant les suites de dérivations élémentaires qui les définissent : l'enchaînement $\beta \xrightarrow{i} \gamma \xrightarrow{j} \delta$, est une dérivation $\beta \xrightarrow{i+j} \delta$.

• La mise d'une règle dans un certain contexte qui nous a servi à définir une dérivation élémentaire, peut s'étendre à toute dérivation : il est facile de définir $\beta_1 \beta \beta_2 \xrightarrow{i} \beta_1 \gamma \beta_2$ en mettant $\beta \xrightarrow{i} \gamma$ dans le contexte (β_1, β_2) .

Exemple 1 (suite).

Reprenons $G = (\mathcal{V}, a + b, R)$ où $\mathcal{V} = S$ et R est défini la règle globale $S \rightarrow SbS + \varepsilon + a + aa$.

Voici une dérivation $S \xrightarrow{11} aabbabaabba$ (comme il a été convenu plus haut, l'occurrence de la variable S à laquelle on applique la règle est soulignée et la règle en question est précisée, pour chaque dérivation élémentaire) :

$$\begin{aligned} (1) \quad \underline{S} &\xrightarrow{1} \underline{S}bS && (S \rightarrow SbS) \\ (2) \quad &\xrightarrow{1} Sb\underline{S}bS && (S \rightarrow SbS) \\ (3) \quad &\xrightarrow{1} \underline{S}bSbSbS && (S \rightarrow SbS) \\ (4) \quad &\xrightarrow{1} SbSb\underline{S}bSbS && (S \rightarrow SbS) \\ (5) \quad &\xrightarrow{1} SbSbSbSb\underline{S}bS && (S \rightarrow SbS) \\ (6) \quad &\xrightarrow{1} SbSbSbSbb\underline{S} && (S \rightarrow \varepsilon) \\ (7) \quad &\xrightarrow{1} \underline{S}bSbSbSbba && (S \rightarrow a) \\ (8) \quad &\xrightarrow{1} aab\underline{S}bSbSbba && (S \rightarrow aa) \\ (9) \quad &\xrightarrow{1} aabb\underline{S}bSbba && (S \rightarrow \varepsilon) \end{aligned}$$

$$(10) \quad \xrightarrow{1} aabbab\underline{S}bba \quad (S \longrightarrow a)$$

$$(11) \quad \xrightarrow{1} aabbabaabba \quad (S \longrightarrow aa)$$

Nous pouvons maintenant définir les deux notions les plus importantes de ce chapitre.

Langages algébriques

Le langage engendré par une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ à partir du mot $\alpha \in (\mathcal{A} + \mathcal{V})^*$ est $\mathcal{L}(G, \alpha) \subseteq \mathcal{A}^*$ défini par

$$u \in \mathcal{L}(G, \alpha) \text{ ssi } \alpha \xrightarrow[G]{*} u$$

pour tout $u \in \mathcal{A}^*$.

Un langage $L \subseteq \mathcal{A}^*$ est dit *algébrique* ssi il existe une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ et $S \in \mathcal{V}$ telles que

$$L = \mathcal{L}(G, S).$$

On est souvent conduit à considérer le langage étendu engendré par G à partir de $\alpha \in (\mathcal{A} + \mathcal{V})^*$, $\mathcal{L}^*(G, \alpha) \subseteq (\mathcal{A} + \mathcal{V})^*$ défini par

$$\beta \in \mathcal{L}^*(G, \alpha) \text{ ssi } \alpha \xrightarrow[G]{*} \beta.$$

On a évidemment $\mathcal{L}(G, \alpha) = \mathcal{L}^*(G, \alpha) \cap \mathcal{A}^*$ mais $\mathcal{L}(G, \alpha) \neq \mathcal{L}^*(G, \alpha)$ en général!

Remarques.

- La seule façon de prouver qu'un langage est algébrique est de montrer l'existence d'une grammaire qui engendre ce langage à partir de l'une de ses variables. Il se peut, bien entendu, que l'on puisse prouver l'existence d'une grammaire sans pour cela être capable de la construire effectivement!

✎ Lorsque l'on étudie G pour elle-même, il n'y a *a priori* aucune raison de privilégier une variable plutôt qu'une autre.

- Lorsque l'on étudie un langage algébrique L , engendré par G , il est nécessaire de préciser la variable $S \in \mathcal{V}$ pour laquelle on a $L = \mathcal{L}(G, S)$: on dit alors que S est l'*axiome* et que le couple (G, S) est une *grammaire avec axiome*. Même dans ce cas, le calcul de $\mathcal{L}(G, S)$ peut nécessiter celui de $\mathcal{L}(G, X)$ pour certaines $X \in \mathcal{V}$: le choix de S dépend du langage que l'on souhaite étudier mais n'est pas imposé par la grammaire elle-même.

1.3 – Les grammaires linéaires.

Avant d'étudier les grammaires dans leur généralité, nous allons considérer des grammaires particulières qui engendrent les langages réguliers, et dont le fonctionnement permet de mettre en évidence une analogie entre les grammaires et les automates. Cette analogie sera moins nette dans le cas général, mais elle sera toujours un bon guide.

Définition.

Une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ est dite *linéaire* lorsque dans chacune de ses règles $X \longrightarrow \alpha$, le mot $\alpha \in (\mathcal{A} + \mathcal{V})^*$ comporte **au plus une variable**. Les règles d'une grammaire linéaire ont donc l'une des formes

$$X \longrightarrow uYv \text{ ou } X \longrightarrow u$$

pour $X \in \mathcal{V}, Y \in \mathcal{V}, u \in \mathcal{A}^*$ et $v \in \mathcal{A}^*$.

Par exemple, la grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ pour laquelle $\mathcal{V} = S + X + Y$, $\mathcal{A} = a + b + c$ et dont les règles globales sont

$$\begin{aligned} S &\longrightarrow aXa + bYb + cSc \\ X &\longrightarrow cY + Sa + b \\ Y &\longrightarrow cX + Sb + a \end{aligned}$$

est linéaire.

La restriction imposée aux règles s'étend aux dérivations : *pour toute variable X et toute dérivation $X \xRightarrow{k} \alpha$ dans une grammaire linéaire, α comporte au plus une variable.*

En fait, ceci implique qu'une telle dérivation est "linéaire" : à chaque étape, au plus une variable est disponible pour appliquer une dérivation élémentaire.

Par exemple, dans la grammaire ci-dessus :

$$\begin{array}{ll}
 S \xRightarrow{1} cSc & (S \longrightarrow cSc) \\
 \xRightarrow{1} caXac & (S \longrightarrow aXa) \\
 \xRightarrow{1} caSaac & (X \longrightarrow Sa) \\
 \xRightarrow{1} cabYbaac & (S \longrightarrow bYb) \\
 \xRightarrow{1} cababaac & (Y \longrightarrow a)
 \end{array}$$

Parmi les grammaires linéaires, les grammaires *linéaires à droite* sont particulièrement intéressantes car elles engendrent les langages réguliers. On qualifie ainsi les grammaires linéaires dont chaque règle est de l'une des formes $X \longrightarrow xY$ ou $X \longrightarrow \varepsilon$ pour $x \in \mathcal{A}$ et $Y \in \mathcal{V}$ (cf. exercice 5).

Propriété

Tout langage régulier est algébrique.

Plus précisément :

Un langage engendré par une grammaire linéaire à droite est régulier, et réciproquement, tout langage régulier est engendré par une grammaire linéaire à droite appropriée.

Grâce au théorème de Kleene, il suffit de vérifier qu'une grammaire linéaire à droite est "équivalente" à un AF.

Les constructions se font facilement en appliquant les correspondances présentées dans le tableau ci-contre.

Grammaire	AF
\mathcal{V}	Q
Axiome	Entrée
$X \rightarrow xY$	$Y \in X \bullet x$
$X \rightarrow \varepsilon$	$X \in F$

- Toute grammaire linéaire à droite engendre des langages réguliers.

Soient $G = (\mathcal{V}, \mathcal{A}, R)$ une grammaire linéaire à droite et $S \in \mathcal{V}$: on va construire un AF $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ à une seule entrée, tel que $\mathcal{L}(\mathbf{A}) = \mathcal{L}(G, S)$.

Pour cela, prenons $Q = \mathcal{V}$: on notera q_X l'état correspondant à la variable X . On peut définir :

- pour toute $X \in \mathcal{V}$, toute $Y \in \mathcal{V}$ et tout $x \in \mathcal{A}$: $q_Y \in \delta(q_X, x)$ ssi $(X \longrightarrow xY)$ est une règle de G ,
- $q_0 = q_S$,
- $q_X \in F$ ssi $X \longrightarrow \varepsilon$ est une règle de G .

La vérification de $\mathcal{L}(\mathbf{A}) = \mathcal{L}(G, S)$ est un exercice élémentaire.

- Réciproquement, tout langage régulier est engendré par une grammaire linéaire à droite à partir de l'une de ses variables.

Soit $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ un AFDC, nous allons construire une grammaire linéaire à droite $G = (\mathcal{V}, \mathcal{A}, R)$ et trouver $S \in \mathcal{V}$ tels que $\mathcal{L}(G, S) = \mathcal{L}(\mathbf{A})$.

Pour cela, on prend $\mathcal{V} = Q$: on notera X_q la variable correspondant à l'état q . On peut définir :

- pour tout $q \in Q$, tout $r \in Q$ et tout $x \in \mathcal{A}$: $X_q \longrightarrow xX_r$ est une règle de G ssi $q \bullet x = r$,
- pour tout $q \in Q$: $X_q \longrightarrow \varepsilon$ est une règle de G ssi $q \in F$,
- $S = X_{q_0}$.

La vérification de $\mathcal{L}(G, S) = \mathcal{L}(\mathbf{A})$ est encore un exercice facile.

Attention : la réciproque de cette propriété est très fautive, en ce sens qu'un langage algébrique n'est que très rarement régulier. Par exemple, $G = (\mathcal{V}, \mathcal{A}, R)$ où $\mathcal{A} = a + b$ et $\mathcal{V} = S$ ne

comporte qu'un seul symbole et dont la règle globale $S \rightarrow aSb + \varepsilon$ engendre évidemment le langage $\mathcal{L}(G, S) = \{a^m b^m \mid m \geq 0\}$ qui, on l'a vu au chapitre précédent, n'est pas régulier!

2 – Représentation arborescente des dérivations.

La définition d'une dérivation comme l'enchaînement linéaire de dérivations a l'avantage d'être séquentielle, mais ceci est généralement obtenu au prix de choix qui, en réalité, sont arbitraires, ou en tout cas, comme nous le verrons dans le chapitre suivant, ne dépendent pas de la dérivation elle-même mais de considérations algorithmiques. Précisons de quels choix arbitraires nous voulons parler.

Dérivations équivalentes.

Soient $Y \rightarrow \beta$ et $Y' \rightarrow \beta'$ deux règles de G . Elles sont, par exemple, applicables à un mot de la forme $\alpha_1 Y \alpha_2 Y' \alpha_3$ pour produire les deux dérivations :

$$\begin{aligned} d_1 : \alpha_1 Y \alpha_2 Y' \alpha_3 &\xrightarrow{1} \alpha_1 \beta \alpha_2 Y' \alpha_3 \xrightarrow{1} \alpha_1 \beta \alpha_2 \beta' \alpha_3, \\ d_2 : \alpha_1 Y \alpha_2 Y' \alpha_3 &\xrightarrow{1} \alpha_1 Y \alpha_2 \beta' \alpha_3 \xrightarrow{1} \alpha_1 \beta \alpha_2 \beta' \alpha_3. \end{aligned}$$

La *commutation* (d'applications consécutives) de règles dans une dérivation est le remplacement de d_1 par d_2 ou celui de d_2 par d_1 dans la dérivation en question : il est clair qu'une telle opération, lorsqu'elle est possible, ne modifie pas la nature d'une dérivation mais seulement la façon de la décrire.

Dérivations équivalentes

Deux dérivations sont dites *équivalentes* ssi on peut transformer l'une en l'autre par une suite de commutations de règles.

La représentation arborescente que nous allons décrire maintenant caractérise l'équivalence entre dérivations de façon plus satisfaisante.

2.1 – Arbres de dérivation d'une grammaire.

L'idée est simplement d'arborer chaque règle $X \rightarrow \xi_1 \dots \xi_m$ de G (c'est-à-dire, de la représenter sous la forme d'un embranchement ordonné, selon le modèle de la figure 1 ci-dessous) et de construire des arbres au lieu de dérivations séquentielles.

Définition des arbres de dérivation.

Dans un arbre

- chaque nœud est étiqueté par un élément de \mathcal{V} et chaque feuille par un élément de $\mathcal{A} + \mathcal{V}$,
- l'embranchement d'un nœud étiqueté par $X \in \mathcal{V}$ (cf. figure 1) est déterminé par une règle $X \rightarrow \alpha$ de G .

Les descendants immédiats d'un tel nœud sont étiquetés par les caractères successifs de α . Remarquez que, dans le cas où $\alpha = \varepsilon$, la représentation est celle d'un nœud qui comporte une suite vide d'embranchement : ε n'est donc pas une feuille, mais l'indication de l'absence définitive de toute feuille!

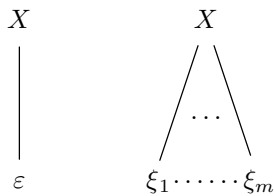


Figure 1.

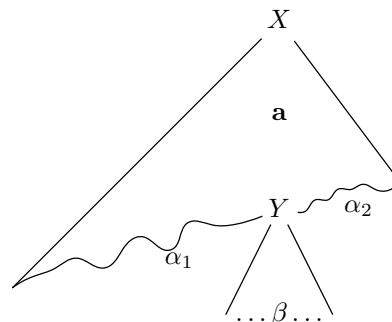


Figure 2.

Nous désignerons par $\mathcal{D}(G)$ l'ensemble des arbres de dérivation de G construits à l'aide ces éléments.

Un arbre $\mathbf{a} \in \mathcal{D}(G)$ sera schématiquement désigné de la façon suivante (volontairement très similaire à celle des dérivations) :

$$\mathbf{a} : \xi \xrightarrow[G]{i} \gamma$$

où

- $\xi \in \mathcal{A} + \mathcal{V}$ est l'étiquette de sa racine,
- $\gamma \in (\mathcal{A} + \mathcal{V})^*$ est sa frondaison (ou frontière), c'est-à-dire, le mot constitué des étiquettes de ses feuilles,
- i est le nombre de ses nœuds.

La grammaire G étant fixée, nous écrirons simplement $\mathbf{a} : \xi \xrightarrow{i} \gamma$.

Une construction inductive de $\mathcal{D}(G)$ est facile à imaginer. Elle utilise deux types particuliers d'arbres :

- *Les arbres triviaux* : tout $\xi \in \mathcal{A} + \mathcal{V}$ définit un arbre $\xi \xrightarrow{0} \xi$ dont la racine est une feuille étiquetée par ξ .
- *Les arbres élémentaires* : une règle $X \rightarrow \alpha$ définit un arbre $X \xrightarrow{1} \alpha$, sur le modèle de la figure 1.

et une opération :

- *La greffe* d'un arbre élémentaire sur la frondaison d'un arbre quelconque : soit un arbre $\mathbf{a} : X \xrightarrow{i} \alpha_1 Y \alpha_2$ et soit $\mathbf{b} : Y \xrightarrow{1} \beta$ une occurrence de l'arbre élémentaire défini par une règle $Y \rightarrow \beta$. Alors on obtient un arbre $\mathbf{c} : X \xrightarrow{i+1} \alpha_1 \beta \alpha_2$ en greffant \mathbf{b} sur la feuille Y de \mathbf{a} , sur le modèle de la figure 2.

2.2 – Arbre d'une dérivation $\xi \xrightarrow{i} \gamma$.

Pour tout $\xi \in \mathcal{A} + \mathcal{V}$, tout $\gamma \in (\mathcal{A} + \mathcal{V})^*$ et toute dérivation $d : \xi \xrightarrow{i} \gamma$, on construit l'arbre de d qui est noté

$$\mathbf{arb}(d) : \xi \xrightarrow{i} \gamma$$

dont le nombre de nœuds i est égal à la longueur de d , par récurrence sur i :

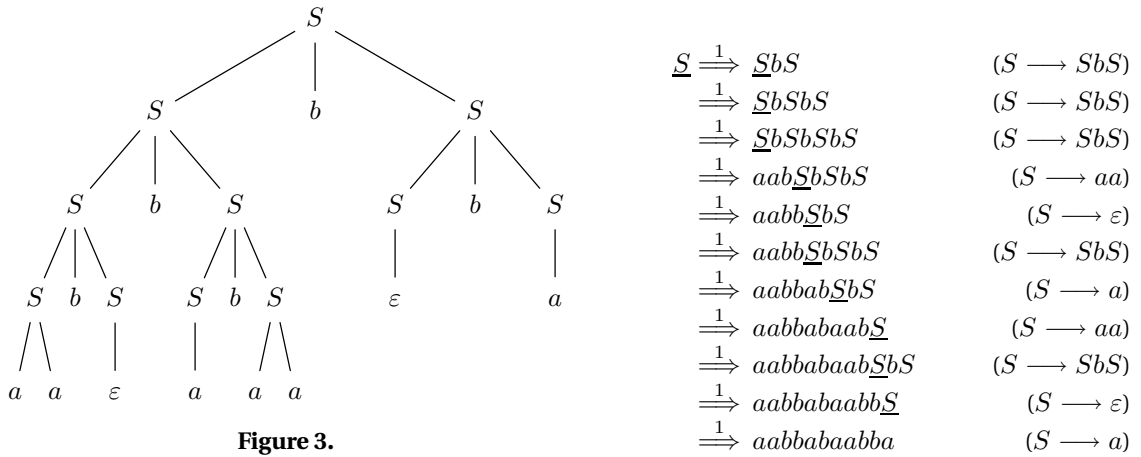
0 : si $d : \xi \xrightarrow{0} \gamma$ alors $\gamma = \xi$ et $\mathbf{arb}(d)$ est trivial.

Dans les autres cas, ξ est nécessairement une variable que nous noterons X .

$i \mapsto i + 1$: si $d : X \xrightarrow{i} \gamma_1 Y \gamma_2 \xrightarrow{1} \gamma_1 \beta \gamma_2$ est la composée de $d' : X \xrightarrow{i} \gamma_1 Y \gamma_2$ et d'une dérivation élémentaire définie par la règle $Y \rightarrow \beta$, $\mathbf{arb}(d)$ s'obtient en greffant l'arbre élémentaire représentant cette règle sur la feuille de $\mathbf{arb}(d')$ étiquetée par l'occurrence en cause de Y (figure 2).

Exemple 1 (suite).

La figure 3 représente l'arbre de la dérivation (de longueur 11) suivante :



2.3 – Arborescence d’une dérivation $\beta \xrightarrow{i} \gamma$.

Une construction inductive de l’ensemble des arborescences de dérivation de G s’obtient en modifiant celle des arbres. Nous désignerons une arborescence par $\mathbf{u} : \beta \xrightarrow{i} \gamma$ où $\beta \in (\mathcal{A} + \mathcal{V})^*$ est le mot constitué des étiquettes de ses racines, $\gamma \in (\mathcal{A} + \mathcal{V})^*$ sa frondaison et i le nombre de ses nœuds. En plus des arbres élémentaires, la construction utilise un type particulier d’arborescence :

- Les arborescences triviales : tout $\beta \in (\mathcal{A} + \mathcal{V})^*$ définit une arborescence $\beta \xrightarrow{0} \beta$.
On remarquera l’existence de l’arborescence $\varepsilon \xrightarrow{0} \varepsilon$.

et une opération de greffe, analogue à la précédente :

- La greffe d’un arbre élémentaire sur la frondaison d’une arborescence quelconque : soit une arborescence $\mathbf{u} : \beta \xrightarrow{i} \alpha_1 Y \alpha_2$ et soit $\mathbf{b} : Y \xrightarrow{1} \beta$ une occurrence de l’arbre élémentaire défini par une règle $Y \rightarrow \alpha$. On obtient ainsi une arborescence $\mathbf{w} : \beta \xrightarrow{i+1} \alpha_1 \alpha \alpha_2$ en greffant \mathbf{b} sur la feuille Y de \mathbf{u} .

La construction de l’arborescence

$$\mathbf{arb}(d) : \beta \xrightarrow{i} \gamma$$

d’une dérivation $d : \beta \xrightarrow{i} \gamma$ est identique à celle qui a été faite plus haut.

Exemple 1 (suite).

Si, dans la dérivation de l’exemple 1, on efface la première dérivation élémentaire, on obtient une dérivation $SbS \xrightarrow{10} aabbabaabba$ dont l’arborescence est représentée par la figure 4.

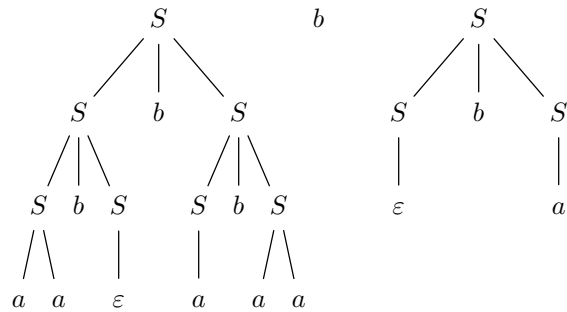


Figure 4.

En fait, la construction inductive des arborescences est celle de juxtapositions d’arbres :

- initialement, tous ces arbres sont triviaux,
- ensuite, chaque greffe se produit sur l’un des arbres de l’arborescence déjà construite.

C’est encore une expression du fait que les grammaires sont indépendantes du contexte. De plus, ceci signifie qu’une arborescence de dérivation de G est un mot sur $\mathcal{D}(G)$:

$$\text{L'ensemble des arborescences de dérivation de } G \text{ est } \mathcal{D}(G)^*.$$

Nous pouvons maintenant caractériser l’équivalence des dérivation de façon géométrique :

Equivalence de dérivations

Deux dérivations sont équivalentes ssi elles ont la même arborescence.

Il suffit essentiellement de vérifier que la commutation de règles, lorsqu'elle est possible, ne modifie pas l'arborescence d'une dérivation. En reprenant les notations du début, la figure 2 bis illustre ce fait sur un arbre : les deux greffes se produisant en deux points distincts de la même frondaison, l'ordre dans lequel on les effectue n'influence pas le résultat.

Exemple.

L'arbre de la figure 3 est celui des deux dérivations des suites de l'exemple 1 : $S \xRightarrow{11} aabbabaabba$ données plus haut. On peut effectivement vérifier qu'elles sont équivalentes.

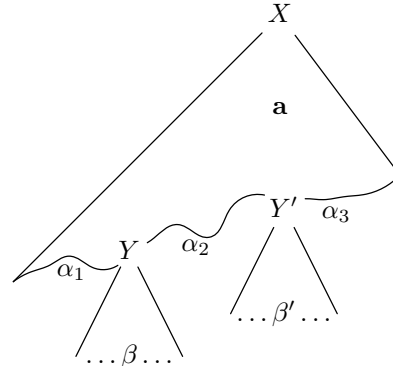


Figure 2 bis.

2.4 – Séquentialisation d'une arborescence.

La *séquentialisation* d'une arborescence $u : \beta \xRightarrow{i} \gamma$ est la construction d'une dérivation $d : \beta \xRightarrow{i} \gamma$ pour laquelle $\text{arb}(d) = u$.

Nous allons voir succinctement les deux types principaux de méthodes de séquentialisation.

Méthode ascendante.

La construction de d se fait par récurrence sur i : la méthode utilisée est dite *ascendante* car la dérivation est construite en partant de la fin.

- si u est triviale alors $\gamma = \beta$ et $d : \beta \xRightarrow{0} \beta$ est une dérivation triviale,
- supposons que la construction soit possible pour toute arborescence à i nœuds et soit $u : \beta \xRightarrow{i+1} \gamma$ à $i + 1$ nœuds; choisissons l'un des nœuds de u qui sont extrémaux :
 - ce nœud et ses embranchements forment l'arbre élémentaire d'une règle $Y \rightarrow \alpha$,
 - l'arborescence u' obtenu en "élagant" les embranchements du nœud choisi, a la forme $\beta \xRightarrow{i} \gamma_1 Y \gamma_2$ où l'étiquette Y est celle du nœud de u qui vient de devenir une feuille de u' , par l'hypothèse de récurrence, c'est l'arborescence d'une dérivation $d' : \beta \xRightarrow{i} \gamma_1 Y \gamma_2$.

Il reste à appliquer une dérivation élémentaire définie avec la règle $Y \rightarrow \alpha$, pour obtenir une dérivation $d : \beta \xRightarrow{i} \gamma_1 Y \gamma_2 \xRightarrow{1} \gamma_1 \alpha \gamma_2$ dont l'arborescence est u .

La vérification de la propriété $\text{arb}(d) = u$ tient au fait que l'élagage qui est utilisé dans la construction de d est l'opération réciproque de la greffe d'un arbre élémentaire utilisée de la construction de $\text{arb}(d)$.

Remarques.

- En général, u admet plusieurs nœuds extrémaux : un choix est donc possible à chaque étape de la construction de d . Tous ces choix produisent des dérivations équivalentes.
- Si $u = \text{arb}(d)$, il est facile de vérifier que d est une des séquentialisations de u : si on numérote les nœuds dans l'ordre où ils se présentent dans la construction de $\text{arb}(d)$, il suffit, à chaque étape de la séquentialisation, d'élaguer le nœud qui porte le plus grand numéro (il est évidemment extrémal au moment où l'on désire élaguer).
- Les algorithmes produits par Yacc, qui nous intéressent particulièrement et que nous étudierons en détail au chapitre suivant, sont basés sur une méthode ascendante.

Méthode descendante.

Dans cette méthode, la dérivation est construite en partant de son début : elle introduit des arborescences même quand on veut séquentialiser un arbre.

Dans l'étape de la récurrence, on élague un embranchement issu de l'une des racines. La figure 5 montre ce qui se passe lorsque l'on applique ce principe à l'arbre de la figure 3 : la partie restante est une arborescence.

La preuve du résultat recherché, qui se fait encore par récurrence sur le nombre de nœuds, est une transposition facile de la précédente.

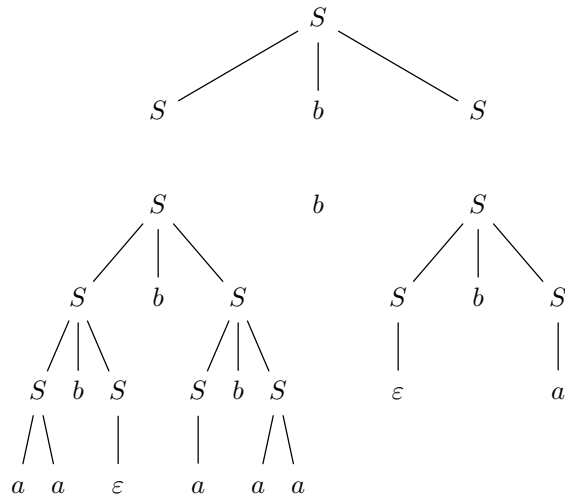


Figure 5.

Remarque.

Les deux méthodes précédentes sont parcimonieuses : elles traitent les nœuds un par un. Dans la pratique, il peut être intéressant de couper une arborescence de façon plus variée : on peut alors appliquer un principe d'induction : une telle méthode est appliquée, par exemple, dans la section 5 ci-dessous, et plus spécialement en 5.3.

3 – Le lemme principal.

On peut résumer tout le début de la présente section en disant qu'une arborescence n'est qu'une autre façon de présenter une dérivation : dans la pratique, on préfère parler de dérivations et manipuler des arborescences! Notons la propriété suivante, qui ne fait, elle aussi, que résumer ce qui vient d'être fait : pour simplifier, nous désignerons $\mathcal{L}(G, \beta)$ par $\mathcal{L}(\beta)$ et $\mathcal{L}^\wedge(G, \beta)$ par $\mathcal{L}^\wedge(\beta)$, jusqu'à la fin de cette section.

Langages engendrés

Pour tout $\gamma \in (\mathcal{A} + \mathcal{V})^*$: $\gamma \in \mathcal{L}^\wedge(\beta)$ ssi $\beta \xrightarrow{*} \gamma$ ssi $\beta \vDash^* \gamma$.

Pour tout $u \in \mathcal{A}^*$: $u \in \mathcal{L}(\beta)$ ssi $\beta \xrightarrow{*} u$ ssi $\beta \vDash^* u$.

Le passage aux arborescences permet de démontrer facilement le lemme principal pour les langages algébriques.

Lemme principal

Soit \mathcal{L} l'application qui à chaque $\alpha \in (\mathcal{A} + \mathcal{V})^*$ associe le langage $\mathcal{L}(\alpha) \subseteq \mathcal{A}^*$ alors :

- 1) \mathcal{L} est un morphisme de monoïdes $\mathcal{L} : ((\mathcal{A} + \mathcal{V})^*, \cdot, \varepsilon) \rightarrow (\mathcal{P}(\mathcal{A}^*), \cdot, \varepsilon)$,
 - 2) \mathcal{L} est (définie par) la substitution $\mathcal{L} : \mathcal{V} \rightarrow \mathcal{P}(\mathcal{A}^*)$.
-

1) Plus généralement, la séquentialisation d'une arborescence "arbre par arbre" permet de vérifier cette propriété pour les langages étendus, c'est-à-dire :

- $\mathcal{L}^\wedge(\varepsilon) = \varepsilon$
- $\mathcal{L}^\wedge(\beta\xi) = \mathcal{L}^\wedge(\beta)\mathcal{L}^\wedge(\xi)$ pour tout $\beta \in (\mathcal{A} + \mathcal{V})^*$ et tout $\xi \in \mathcal{A} + \mathcal{V}$.

2) Par la propriété principale de $(\mathcal{A} + \mathcal{V})^*$ (cf. chapitre 1, section 5.2) ceci signifie que \mathcal{L} est entièrement déterminée par sa restriction aux éléments de $\mathcal{A} + \mathcal{V}$ et, en se rappelant que $\mathcal{L}(x) = x$ pour tout $x \in \mathcal{A}$, par sa restriction $\mathcal{L} : \mathcal{V} \rightarrow \mathcal{P}(\mathcal{A}^*)$ aux variables.

Compte tenu des résultats du chapitre 1, les deux propriétés précédentes sont équivalentes.

Remarque.

On peut préciser un peu la propriété 1) en vue de son application pratique.

L'arborescence d'une dérivation $d : \beta\xi \xrightarrow{k} \gamma'$ est un élément de $\mathcal{D}(G)^*$ qui se présente sous la forme $ua : \beta\xi \xrightarrow{k} \gamma\delta$ où $\gamma\delta = \gamma'$ et :

- $u : \beta \xrightarrow{i} \gamma$ est l'arborescence d'une dérivation $g : \beta \xrightarrow{i} \gamma$,
- $a : \xi \xrightarrow{j} \delta$ est l'arbre d'une dérivation $h : \xi \xrightarrow{j} \delta$,

avec, bien entendu $i + j = k$.

A partir de cette décomposition, on peut reconstituer, non pas d elle-même mais, des dérivations équivalentes à d , savoir :

- la composée de $g_1 : \beta\xi \xrightarrow{i} \gamma\xi$ et de $h_1 : \gamma\xi \xrightarrow{j} \gamma\delta$, dans cet ordre,
 - ou
 - la composée de $h_2 : \beta\xi \xrightarrow{j} \beta\delta$ et de $g_2 : \beta\delta \xrightarrow{i} \gamma\delta$, dans cet ordre,
- dont l'arborescence est évidemment ua .

Exemple 1 (suite).

Regardons le cas de la dérivation $d : SbS \xrightarrow{10} aabbabaabba$ dont l'arborescence est à la figure 4 ci-dessus :


$$\begin{aligned} d_1 : S &\xrightarrow{7} aabbabaa \\ d_2 : b &\xrightarrow{0} b \\ d_3 : S &\xrightarrow{3} ba \end{aligned}$$

sont des séquentialisations respectives des trois arbres de cette figure 4.

On peut construire une dérivation équivalente à d , en composant, par exemple

$$\begin{aligned} d'_1 : SbS &\xrightarrow{7} aabbabaabS \\ d'_2 : aabbabaabS &\xrightarrow{0} aabbabaabS \\ d'_3 : aabbabaabS &\xrightarrow{3} aabbabaabba \end{aligned}$$

dans cet ordre.

 e lemme principal est rarement cité explicitement car il est la chose la plus naturelle que l'on puisse dire à propos d'une grammaire indépendante du contexte. Cependant, on peut indiquer son usage par des expressions comme :

- $d : \beta \xrightarrow{k} \gamma$ se décompose en des $d_j : \eta_j \xrightarrow{k_j} \gamma_i, \dots$
- les $d_j : \eta_j \xrightarrow{k_j} \gamma_i$ se recomposent en $\eta_1 \dots \eta_n \xrightarrow{k} \gamma_1 \dots \gamma_n \dots$

(les points de suspension signalent que le mot β est supposé connu de façon explicite). Nous en verrons des exemples par la suite : notons simplement que le lemme principal permet de faire des raisonnements par induction lorsque les k_j sont strictement inférieurs à k .

3.1 – Système d'équations en langages associé à une grammaire.

Soit $G = (\mathcal{V}, \mathcal{A}, R)$ une grammaire. Nous avons constaté au début de ce chapitre que G était équivalente à la donnée de la substitution l définissant les seconds membres des règles globales de G . Il est donc possible d'appliquer à G les méthodes et les résultats exposés dans le chapitre 1, section 6.6, mais nous allons surtout utiliser les moyens propres au présent chapitre.

A cet effet, étendons l aux constantes par $l(x) = x$ pour tout $x \in \mathcal{A}$. Alors, assimilant les variables à des inconnues (en langages sur \mathcal{A}), l'ensemble des équations

$$X = l(X) \text{ pour toute } X \in \mathcal{V}$$

est appelé **le système d'équations associé à G** .

Une solution d'un tel système est la donnée, pour chaque $X \in \mathcal{V}$, d'une valeur $s(X) \subseteq \mathcal{A}^*$, de telle façon que ces valeurs vérifient les égalités que le système exprime : c'est donc une substitution $s : \mathcal{V} \rightarrow \mathcal{P}(\mathcal{A}^*)$ vérifiant l'égalité

$$s(X) = s(\mathbf{l}(X)) \text{ pour chaque } X \in \mathcal{V}$$

(pour écrire ces égalités, on étend s à tous les symboles en posant $s(x) = x$ pour tout $x \in \mathcal{A}$).

Le lemme principal, sous sa forme 2), nous permet d'écrire :

Résolution du système d'équations associé à G

La substitution $\mathcal{L} : \mathcal{V} \rightarrow \mathcal{P}(\mathcal{A}^)$ est la plus petite solution du système associé à la grammaire G .*

Montrons d'abord que c'est une solution, c'est-à-dire que

$$\mathcal{L}(X) = \mathcal{L}(\mathbf{l}(X)) \text{ pour toute } X \in \mathcal{V}.$$

Les équivalences suivantes, qui sont des applications simples des définitions :

$$\begin{aligned} u \in \mathcal{L}(X) & \text{ ssi } X \xrightarrow{*} u \\ & \text{ssi il existe } \alpha \in (\mathcal{A} + \mathcal{V})^* \text{ tel que } (X \rightarrow \alpha) \in R \text{ et } \alpha \xrightarrow{*} u \\ & \text{ssi il existe } \alpha \in \mathbf{l}(X) \text{ tel que } \alpha \xrightarrow{*} u \\ & \text{ssi } u \in \mathcal{L}(\mathbf{l}(X)) \end{aligned}$$

sont vraies pour tout $u \in \mathcal{A}^*$: on a donc bien l'égalité annoncée.

Montrons maintenant que c'est la plus petite :

soit $M : \mathcal{V} \rightarrow \mathcal{P}(\mathcal{A}^*)$ une solution quelconque du système, il faut montrer que $\mathcal{L}(X) \subseteq M(X)$ pour toute $X \in \mathcal{V}$. L'hypothèse sur M signifie que l'on a toujours $M(X) = M(\mathbf{l}(X))$, en particulier $M(\mathbf{l}(X)) \subseteq M(X)$, c'est-à-dire

$$M(\alpha) \subseteq M(X)$$

pour tout $\alpha \in \mathbf{l}(X)$.

Montrons, pour tout $u \in \mathcal{L}(X)$, par induction sur la longueur des dérivations $X \xrightarrow{1} \alpha \xrightarrow{k} u$, que $u \in M(X)$:

- si $k = 0$ alors $\alpha = u$ donc $u = M(u) = M(\alpha) \subseteq M(X)$,
- sinon, le lemme principal permet de décomposer $\alpha \xrightarrow{k} u$ en dérivations de longueurs strictement inférieures à $k + 1$: l'hypothèse d'induction permet alors de déduire que $u \in M(\alpha)$, et donc que $u \in M(X)$.

Ceci termine la démonstration de la propriété.

Exemple 1 (suite).

Le système de la grammaire de notre exemple est réduit à l'unique équation $S = SbS + \varepsilon + a + aa$, vérifions que $\mathcal{L}(S)$. En appliquant le lemme principal, on peut écrire :

$$\begin{aligned} \mathcal{L}(\mathbf{l}(S)) &= \mathcal{L}(SbS + \varepsilon + a + aa) \\ &= \mathcal{L}(SbS) + \mathcal{L}(\varepsilon) + \mathcal{L}(a) + \mathcal{L}(aa) \\ &= \mathcal{L}(S)b\mathcal{L}(S) + \varepsilon + a + aa \end{aligned}$$

L'égalité $\mathcal{L}(S) = \mathcal{L}(\mathbf{l}(S))$ signifie donc bien que $\mathcal{L}(S) = \mathcal{L}(S)b\mathcal{L}(S) + \varepsilon + a + aa$.

Remarque.

On a vu dans la section 6.6 du chapitre 1 que la plus petite solution du système est $s = c \circ (id + v)^*$ où $c(X) = \mathbf{l}(X) \cap \mathcal{A}^*$ et $v(X) = \mathbf{l}(X) - c(X)$, on a donc


$$\mathcal{L} = c \circ (id + v)^*.$$

- Pour illustrer cette égalité, il suffit de constater qu'une dérivation $X \xrightarrow{k} u$ de $u \in \mathcal{A}^*$ à partir de $X \in \mathcal{V}$ est équivalente à la composée de deux dérivations :

- $X \xrightarrow{i} \beta$ qui n'applique que des règles du type $Y \longrightarrow \alpha$ pour $\alpha \in v(Y)$,
- $\beta \xrightarrow{j} u$ qui n'applique que des règles du type $Y \longrightarrow v$ pour $v \in c(Y)$.

Si donc $u \in \mathcal{L}(X)$, il existe $\beta \in (id + v)^*(X)$ tel que $u \in c(\beta)$ et réciproquement.

• La méthode qui vient d'être évoquée consiste à faire une construction par récurrence de la plus petite solution du système, ce qui, dans beaucoup de cas est parfaitement suffisant pour décider si un mot de longueur donnée appartient ou non au langage engendré (c'est le cas pour les grammaires propres, dont on fera l'étude par la suite).

 n fait, caractériser de façon "intéressante" les langages engendrés par une grammaire particulière est toujours un problème particulier (et intéressant)!

3.2 – Définitions inductives.

Comme on l'a déjà observé à deux reprises dans le chapitre 1 (sections 4 et 6.6) la plus petite solution que l'on vient d'obtenir est aussi la plus petite solution du système d'inéquations associé à G :

$$I(X) \subseteq X.$$

On peut énoncer cette propriété sous la forme d'une définition inductive simultanée des langages $\mathcal{L}(X)$. Par exemple, dans le cas d'une grammaire ne comportant qu'une seule variable :

$\mathcal{L}(X)$ est le plus petit langage

- qui contient des éléments donnés (ceux de $c(X)$),
- qui est stable par des opérations données (celles décrites par les éléments de $v(X)$).

Cette remarque générale donne lieu à de multiples applications.

Exemple 1 (suite).

Soit $L \subseteq (a + b)^*$ le langage constitué des mots qui ne comportent pas le facteur aaa : il est facile de vérifier que ce langage est régulier mais, nous allons l'engendrer par une grammaire (qui n'est pas linéaire) traduisant une définition inductive très naturelle de L .

- Les deux propriétés suivantes sont immédiates :

$$\begin{array}{ll} 1) \varepsilon \in L, a \in L \text{ et } aa \in L, & (\varepsilon + a + aa \subseteq L) \\ 2) \text{ si } u_1 \in L \text{ et } u_2 \in L \text{ alors } u_1 b u_2 \in L & (L b L \subseteq L) \end{array}$$

- Tous les éléments de L sont obtenus de cette façon : soit $u \in L$

- si $|u|_b = 0$ alors $u \in \varepsilon + a + aa$;
- sinon, en choisissant une occurrence quelconque de b dans u on obtient une décomposition $u = u_1 b u_2$ où $u_1 \in L$ et $u_2 \in L$.

ce qui signifie bien que L est le plus petit langage sur l'alphabet $\mathcal{A} = a + b$ vérifiant les propriétés 1) et 2) : on vient donc de donner une définition inductive de L et ceci est suffisant pour prouver que L est le langage engendré par la grammaire dont l'unique règle globale est $S \longrightarrow S b S + \varepsilon + a + aa$.

Exemple 2.

Cet exemple est classique (et c'est un cas particulier de celui qui est proposé dans l'exercice 1.22).

Soit $\mathcal{A} = \{p, q, r, \neg, \wedge, \vee, \rightarrow, [,]\}$ où p, q et r sont des "variables propositionnelles", \neg, \wedge, \vee et \rightarrow des symboles logiques, $[$ et $]$ des parenthèses. On définit l'ensemble F des propositions sur p, q et r par les propriétés suivantes :

$$\begin{array}{ll} \text{a) } p \in F, q \in F \text{ et } r \in F, & (p + q + r \subseteq F) \\ \text{b) si } A \in F \text{ alors } \neg A \in F, & (\neg F \subseteq F) \\ \text{c) si } A \in F \text{ et } B \in F \text{ alors } [A \wedge B] \in F, [A \vee B] \in F \text{ et } [A \rightarrow B] \in F, & ([F \wedge F] \subseteq F, \dots) \end{array}$$

et tout élément de F est obtenu ainsi.

Ceci constitue une définition inductive de F . Soit donc $G = (\mathcal{V}, \mathcal{A}, R)$ où $\mathcal{V} = S$ n'a qu'un seul élément, \mathcal{A} est comme ci-dessus et R est défini par la règle globale

$$S \longrightarrow \neg S + [S \wedge S] + [S \vee S] + [S \rightarrow S] + p + q + r.$$

alors $F = \mathcal{L}(G, S)$.

4 – Transformations des grammaires.

On se souvient qu'un langage algébrique $L \subseteq \mathcal{A}^*$ est un langage engendré par une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ à partir d'une variable (appelée "axiome") $S \in \mathcal{V}$, c'est-à-dire $L = \mathcal{L}(G, S)$.

- G définit un procédé de calcul : l'application des règles permet de produire chaque mot $u \in L$ comme résultat d'une dérivation $S \xrightarrow[G]{k} u$.

- Réciproquement, si $u \in \mathcal{A}^*$, la question se pose de savoir si $u \in L$: une réponse affirmative ne sera satisfaisante que si elle est accompagnée de sa preuve, c'est-à-dire, d'une dérivation $S \xrightarrow[G]{k} u$. De même, une réponse négative ne sera satisfaisante que si l'on prouve que toute tentative de construire une telle dérivation échoue.

Ces deux questions, synthèse (*synthesis*) et analyse (*parsing*) syntaxiques, justifient que l'on s'intéresse à des grammaires de formes particulières : il s'agit alors de voir s'il est possible de transformer une grammaire donnée en une grammaire de la forme particulière souhaitée, qui soit capable d'engendrer le même langage. Nous allons en voir quelques unes, mais ce petit catalogue n'est qu'un modeste début.

Dans tout ce qui suit, G désignera une grammaire $(\mathcal{V}, \mathcal{A}, R)$.

4.1 – Grammaires réduites.

A. Restriction de l'alphabet des variables.

Soit $\mathcal{V}' \subseteq \mathcal{V}$: la restriction de G à \mathcal{V}' est la grammaire $G' = (\mathcal{V}', \mathcal{A}, R')$ où $R' = R \cap (\mathcal{V}' \times (\mathcal{A} + \mathcal{V}')^*)$ est l'ensemble des règles de G qui peuvent s'écrire avec des variables de \mathcal{V}' :

$$(X \rightarrow \alpha) \in R' \text{ ssi } (X \rightarrow \alpha) \in R, X \in \mathcal{V}' \text{ et } \alpha \in (\mathcal{A} + \mathcal{V}')^*.$$

De façon évidente $\mathcal{L}(G', X) \subseteq \mathcal{L}(G, X)$ (et plus généralement $\mathcal{L}^*(G', X) \subseteq \mathcal{L}^*(G, X)$) pour toute $X \in \mathcal{V}'$, car toute règle de G' est aussi une règle de G .

Remarque.

Toutes les parties de \mathcal{V} que nous serons amenés à définir par la suite sont construites de la façon suivante : On définit une suite \mathcal{U}_i croissante de parties de \mathcal{V} :

$$\mathcal{U}_0 \subseteq \mathcal{U}_1 \subseteq \dots \subseteq \mathcal{U}_i \subseteq \dots \subseteq \mathcal{V}.$$

Une telle suite est stationnaire car \mathcal{V} est fini, plus précisément : il existe un entier N tel que $i \geq N$ implique $\mathcal{U}_i = \mathcal{U}_N$. $\mathcal{U} = \mathcal{U}_N$ s'appelle la limite de la suite \mathcal{U}_i .

B. Variables productives.

- $X \in \mathcal{V}$ est productive ssi $\mathcal{L}(G, X) \neq \emptyset$. $\text{Prod}(G) \subseteq \mathcal{V}$ désignera l'ensemble des variables productives de G .

Elimination des variables non productives

La restriction $G' = (\mathcal{V}', \mathcal{A}, R')$ de G à $\mathcal{V}' = \text{Prod}(G)$ vérifie

- toutes ses variables sont productives
 - $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}'$.
-

Le deuxième point implique le premier, il suffit donc de montrer que $\mathcal{L}(G, X) \subseteq \mathcal{L}(G', X)$ pour $X \in \text{Prod}(G)$. Soit $d : X \xrightarrow[G]{k} u$ pour $u \in \mathcal{A}^*$, nous allons montrer que d est aussi une dérivation dans G' , c'est-à-dire qu'elle ne fait intervenir que des variables productives; en effet, si Y intervient dans d , celle-ci peut s'écrire comme une composée $X \xrightarrow[G]{l} \alpha Y \beta \xrightarrow[G]{m} u$: le lemme principal nous fournit alors une dérivation $Y \xrightarrow[G]{m_X} v$ où v est un facteur de u , Y est donc productive.

$$\text{Prod}(G)$$

$\mathcal{P}rod(G)$ est égal à la limite \mathcal{U} de la suite définie par :

$$\mathcal{U}_0 = \emptyset \quad \mathcal{U}_{i+1} = \mathcal{U}_i + \{X \in \mathcal{V} \mid \text{il existe } (X \longrightarrow \alpha) \in R \text{ telle que } \alpha \in (\mathcal{A} + \mathcal{U}_i)^*\}.$$

Il faut montrer que $\mathcal{U} = \mathcal{P}rod(G)$:

- $\mathcal{U} \subseteq \mathcal{P}rod(G)$. Montrons que $\mathcal{U}_i \subseteq \mathcal{P}rod(G)$ pour tout i :
 - c'est vrai pour 0!
 - supposons que ce soit vrai pour i et soit $X \in \mathcal{U}_{i+1}$: si $X \in \mathcal{U}_i$, il n'y a rien à prouver; sinon on a $\alpha \in (\mathcal{A} + \mathcal{U}_i)^*$ et $X \xrightarrow[G]{1} \alpha$: chaque variable de α étant dans \mathcal{U}_i , l'hypothèse de récurrence donne une dérivation de chacune d'entre elles en un facteur de u , que le lemme principal permet de recomposer en $\alpha \xrightarrow[G]{k} u$, d'où une dérivation $X \xrightarrow[G]{k+1} u$ dont l'existence signifie que $X \in \mathcal{P}rod(G)$.
- $\mathcal{P}rod(G) \subseteq \mathcal{U}$. Il faut montrer que si $X \in \mathcal{P}rod(G)$ (c'est-à-dire que si l'on a $d : X \xrightarrow[G]{k+1} u$ pour un $u \in \mathcal{A}^*$) alors il existe i tel que $X \in \mathcal{U}_i$: ceci se fait naturellement par induction sur la longueur des dérivations. d est la composée d'une dérivation élémentaire $X \xrightarrow[G]{1} \alpha$ et de $\alpha \xrightarrow[G]{k} u$. La décomposition de cette dernière nous permet de trouver une dérivation $Y \xrightarrow[G]{k_Y} v$ de longueur $k_Y \leq k < k+1$ pour chaque occurrence de variable Y de α donc, grâce à l'hypothèse d'induction, il existe j tel que $Y \in \mathcal{U}_j$. Soit j_0 le plus grand des j ainsi trouvés : toutes les variables de α sont des éléments de \mathcal{U}_{j_0} (puisque la suite des \mathcal{U}_i est croissante) donc $\alpha \in (\mathcal{A} + \mathcal{U}_{j_0})^*$, c'est-à-dire $X \in \mathcal{U}_{j_0+1}$.

Cette description de $\mathcal{P}rod(G)$ fournit un algorithme pour tester la vacuité d'un langage de la forme $\mathcal{L}(G, X)$.

C. Variables accessibles depuis une variable donnée.

- $X \in \mathcal{V}$ est *accessible depuis* $S \in \mathcal{V}$ ssi $\mathcal{L}(G, S) \cap [X] \neq \emptyset$, où $[X] \subseteq (\mathcal{A} + \mathcal{V})^*$ est l'ensemble des α tels que $|\alpha|_X > 0$. $\mathcal{A}cc_G(S) \subseteq \mathcal{V}$ désignera l'ensemble des variables accessibles à partir de S .

On a $S \in \mathcal{A}cc_G(S)$ grâce à la dérivation triviale $S \xrightarrow[G]{0} S$.

Partie accessible depuis une variable

Soit $S \in \mathcal{V}$ telle que $\mathcal{L}(G, S) \neq \emptyset$.

La restriction $G' = (\mathcal{V}', \mathcal{A}, R')$ de G à $\mathcal{V}' = \mathcal{A}cc_G(S)$ vérifie

- toute $X \in \mathcal{V}'$ est accessible à partir de S
- $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}'$.

Pour la preuve, prenons $X \in \mathcal{A}cc_G(S)$.

Toute variable intervenant dans une dérivation $d : S \xrightarrow[G]{k} \alpha X \beta$ est accessible à partir de S , d est donc une dérivation dans G' : le premier point est vérifié. Le second est tout aussi évident : si $e : X \xrightarrow[G]{l} u$, toutes les variables de la dérivation composée $S \xrightarrow[G]{k+l} \alpha u \beta$ sont accessibles à partir de S , cette dernière est donc une dérivation dans G' , donc e l'est aussi.

$\mathcal{A}cc_G(S)$

$\mathcal{A}cc_G(S)$ est égal à la limite \mathcal{U} de la suite définie par :

$$\mathcal{U}_0 = S \quad \mathcal{U}_{i+1} = \mathcal{U}_i + \{X \in \mathcal{V} \mid \text{il existe } Y \in \mathcal{U}_i \text{ et } \alpha \in [X] \text{ t.q. } (Y \longrightarrow \alpha) \in R\}.$$

Il faut vérifier que $\mathcal{U} = \mathcal{A}cc_G(S)$, ce qui est un exercice facile.

D. Grammaires réduites.

- G est réduite pour $S \in \mathcal{V}$ ssi toutes ses variables sont productives et accessibles à partir de S .

Réduction des grammaires

Pour toute $S \in \mathcal{V}$ telle que $\mathcal{L}(G, S) \neq \emptyset$ il existe une grammaire $G' = (\mathcal{V}', \mathcal{A}, R')$ telle que

- $S \in \mathcal{V}' \subseteq \mathcal{V}$,
 - G' est réduite pour S ,
 - $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}'$.
-

Il suffit d'éliminer les variables non productives puis les variables non accessibles.

Attention.

⚠ L'élimination des variables qui ne sont pas productives peut rendre d'autres variables inaccessibles à partir de S , il faut donc effectuer les opérations d'élimination dans cet ordre pour réduire G . Pour s'en rendre compte, on pourra éliminer d'abord les variables non accessibles depuis S , puis les variables non productives, dans la grammaire sur $\mathcal{V} = S + A + B$ et $\mathcal{A} = a + b$, dont les règles globales sont $S \rightarrow A + a$, $A \rightarrow AB$ et $B \rightarrow b$, et d'observer le résultat!

La propriété pour une grammaire G d'être réduite n'a pas une importance théorique très considérable, d'autant qu'elle est relative à une variable S . Par contre, il y a un intérêt pratique évident à éliminer les variables qui ne sont pas productives : par la suite, nous supposons toujours que notre grammaire G ne comporte que des variables productives.

4.2 – Grammaires propres.

Dans les transformations précédentes, les variables jouaient le premier rôle, et l'ensemble des règles n'était modifié que par la force des choses. Dans les transformations que nous allons voir maintenant, les règles seront modifiées pour des raisons plus profondes.

E. Production de ε .

- G produit ε ssi il existe $X \in \mathcal{V}$ tel que $\varepsilon \in \mathcal{L}(G, X)$.
- L'ensemble $\mathcal{Eps}(G) \subseteq \mathcal{V}$ des variables produisant ε est défini par $X \in \mathcal{Eps}(G)$ ssi $\varepsilon \in \mathcal{L}(G, X)$.

Elimination des productions de ε

Il existe une grammaire $G' = (\mathcal{V}, \mathcal{A}, R')$ qui ne produit pas ε et qui vérifie $\mathcal{L}(G', X) = \mathcal{L}(G, X) - \varepsilon$ pour toute $X \in \mathcal{V}$.

Construction de G' .

- Considérons la substitution $s : \mathcal{A} + \mathcal{V} \rightarrow \mathcal{P}((\mathcal{A} + \mathcal{V})^*)$ définie par $s(\xi) = \begin{cases} \xi + \varepsilon & \text{si } \xi \in \mathcal{Eps}(G), \\ \xi & \text{sinon.} \end{cases}$
- Si la règle globale de X dans G est $X \xrightarrow{G} \mathbf{1}$, alors sa règle globale dans G' est $X \xrightarrow{G'} s(\mathbf{1}) - \varepsilon$.

Chaque variable $X \in \mathcal{Eps}(G)$ se trouve donc remplacée par $X + \varepsilon$: on ne peut pas se contenter de supprimer les règles de la forme $X \rightarrow \varepsilon$ car celles-ci peuvent intervenir dans des dérivations qui aboutissent à des $u \neq \varepsilon$! Considérons par exemple, la grammaire $G = (S + A, a + b, R)$ dont les règles globales sont :

$$\begin{aligned} S &\rightarrow SAS + b \\ A &\rightarrow a + \varepsilon \end{aligned}$$

Grâce à l'existence de la dérivation $\underline{S} \xrightarrow{1} \underline{SAS} \xrightarrow{1} \underline{bAS} \xrightarrow{1} \underline{bS} \xrightarrow{1} \underline{bb}$, on a $bb \in \mathcal{L}(G, S)$, mais il est clair que si on modifie G en G' en éliminant simplement la règle $A \rightarrow \varepsilon$, alors $bb \notin \mathcal{L}(G', S)$! Ceci, comme la démonstration le confirmera, justifie la définition.

Exemple 2.

Soit $G = (S + A + B, a + b, R)$ la grammaire définie par les règles globales :

$$\begin{aligned} S &\longrightarrow aAB + BA + b \\ A &\longrightarrow BBB + a \\ B &\longrightarrow AB + b + \varepsilon \end{aligned}$$

Il est facile de constater que $\mathcal{Eps}(G) = S + A + B$, les règles globales de G' s'écrivent donc :

$$\begin{aligned} S &\longrightarrow (a(A + \varepsilon)(B + \varepsilon) + (B + \varepsilon)(A + \varepsilon) + b) - \varepsilon \\ A &\longrightarrow ((B + \varepsilon)(B + \varepsilon)(B + \varepsilon) + a) - \varepsilon \\ B &\longrightarrow ((A + \varepsilon)(B + \varepsilon) + b + \varepsilon) - \varepsilon \end{aligned}$$

c'est-à-dire, en développant les seconds membres :

$$\begin{aligned} S &\longrightarrow aAB + aA + aB + a + BA + A + B + b \\ A &\longrightarrow BBB + BB + B + a \\ B &\longrightarrow AB + A + B + b \end{aligned}$$

Démonstration de la propriété.

G' ne produit pas ε puisqu'elle n'a aucune règle de la forme $X \xrightarrow{G'} \varepsilon$, il reste à prouver la propriété relative aux langages.

- $\mathcal{L}(G', X) \subseteq \mathcal{L}(G, X) - \varepsilon$.

Soit $d' : X \xrightarrow{G'}^{k'+1} u$, on a $u \in \mathcal{A}^* - \varepsilon$ et on doit construire $d : X \xrightarrow{G}^{k+1} u$: ceci se fait par induction.

- Soient $\rho' : X \xrightarrow{G'} \alpha'$ la première règle de d' et $\rho : X \xrightarrow{G} \alpha$ une règle de G dont ρ' provient par la transformation précédente : α' est obtenu en substituant ε à certaines occurrences de variables $Y \in \mathcal{Eps}(G)$ de α . Pour chaque $Y \in \mathcal{Eps}(G)$ on a une dérivation $\varepsilon_Y : Y \xrightarrow{G} \varepsilon$.

- Soit $e' : \alpha' \xrightarrow{G'}^{k'} u$ la partie de d' obtenue en amputant celle-ci de ρ' .

- si $k' = 0$, $\alpha' = u$ et d est la composée de la dérivation élémentaire définie par $\rho : X \xrightarrow{G} \alpha$ et de la dérivation $\alpha \xrightarrow{G}^k u$, recomposée à partir des ε_Y convenables.

- Sinon, par décomposition de e' , on dispose d'une dérivation $e'_Z : Z \xrightarrow{G'}^{k'_Z} v_Z$ pour chaque variable Z de α' , à laquelle on peut appliquer l'hypothèse d'induction puisque $k'_Z \leq k' < k' + 1$: on obtient ainsi $e_Z : Z \xrightarrow{G}^{k_Z} v_Z$. Maintenant, les e_Z et les ε_Y permettent de construire une dérivation $e : \alpha \xrightarrow{G}^k u$: d est la composée de la dérivation élémentaire définie par ρ et de e .

- $\mathcal{L}(G, X) - \varepsilon \subseteq \mathcal{L}(G', X)$.

Cette démonstration se fait encore par induction mais, cette fois, il s'agit de faire disparaître les dérivations qui produisent ε : un excellent exercice.

$\mathcal{Eps}(G)$

$\mathcal{Eps}(G)$ est la limite \mathcal{U} de la suite définie par :

$$\mathcal{U}_0 = \emptyset \quad \mathcal{U}_{i+1} = \mathcal{U}_i + \{X \in \mathcal{V} \mid \text{il existe } \alpha \in \mathcal{U}_i^* \text{ tel que } (X \longrightarrow \alpha) \in R\}.$$

Il faut se souvenir que $\emptyset^* = \varepsilon!$

F. ε -dérivations.

- Une ε -dérivation est une dérivation de la forme $X \xrightarrow{G}^k Y$ où $X \in \mathcal{V}$, $Y \in \mathcal{V}$ et $k > 0$.
- La clôture d'une variable X est définie par $\mathcal{Cl}_G(X) = \{Y \in \mathcal{V} \mid X \xrightarrow{G}^* Y\}$.

On a $X \in Cl_G(X)$ grâce à l'existence de la dérivation triviale $X \xrightarrow[G]{0} X$.

Par la suite, on notera simplement $Cl(X)$ pour $Cl_G(X)$.

- Pour chaque ensemble de la forme $Cl(X)$ on introduit une variable, désignée par \bar{X} : vous noterez que \bar{X} et \bar{Y} sont égales lorsque $Cl(X) = Cl(Y)$.

- Si $\bar{\mathcal{V}}$ désigne l'ensemble de ces variables, on a une application $c : \mathcal{V} \rightarrow \bar{\mathcal{V}}$, définie par $c(X) = \bar{X}$.

Nous supposons que G ne produit pas ε car alors, une ε -dérivation est composée uniquement d' ε -dérivations élémentaires $Z \xrightarrow[G]{1} T$, et c'est seulement dans ce cas que la notion est pleinement significative. On devra donc, si besoin est, commencer par appliquer la transformation précédente à la grammaire G , pour lui interdire de produire ε .

Élimination des ε -dérivations

Soit G une grammaire ne produisant pas ε alors, il existe une grammaire $\bar{G} = (\bar{\mathcal{V}}, \mathcal{A}, \bar{R})$ qui n'admet pas d' ε -dérivation, telle que $\mathcal{L}(\bar{G}, \bar{X}) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}$.

La construction des règles globales de \bar{G} suit la leçon de la détermination des ε -AF (cf. chapitre 2, section 5.2).

Soit $X \xrightarrow[G]{1} I(X)$ la règle globale pour $X \in \mathcal{V}$ et considérons $\mathbf{m}(X) = I(X) - \mathcal{V}$ l'ensemble des éléments de $I(X)$ qui ne se réduisent pas à une seule variable, alors la règle globale pour $c(X)$ dans \bar{G} est

$$c(X) \longrightarrow c(\mathbf{m}(Cl(X))).$$

Dans cette définition, la substitution \mathbf{m} a été étendue aux langages, comme d'habitude : $\mathbf{m}(Cl(X))$ désigne donc la réunion des $\mathbf{m}(Y)$ lorsque Y parcourt $Cl(X)$. De même, c est une substitution qui remplace chaque occurrence d'une variable $X \in \mathcal{V}$ par la nouvelle variable $\bar{X} \in \bar{\mathcal{V}}$.

Si l'on ne modifie pas les variables (par l'application de c), la construction précédente consiste à considérer les règles globales

$$X \longrightarrow \mathbf{m}(Cl(X))$$

qui créent une règle $X \longrightarrow \beta$ pour chaque dérivation $X \xrightarrow[G]{k} Y \xrightarrow[G]{1} \beta$ où $Y \in \mathcal{V}$ et $\beta \notin \mathcal{V}$. La grammaire G' ainsi obtenue n'a plus d' ε -dérivation et vérifie $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ de façon évidente. On applique c pour prendre en compte le fait que deux variables vérifiant $Cl(X) = Cl(Y)$ ont la même règle globale dans G' .

La clôture d'une variable se calcule effectivement de la façon suivante.

$$Cl_G(X)$$

Soit G une grammaire ne produisant pas ε alors, $Cl_G(X)$ est la limite \mathcal{U} de la suite définie par :

$$\mathcal{U}_0 = X \quad \mathcal{U}_{i+1} = \mathcal{U}_i + \{Z \in \mathcal{V} \mid \text{il existe } Y \in \mathcal{U}_i \text{ tel que } (Y \longrightarrow Z) \in R\}.$$

Exemple 2 (suite).

Reprenons la grammaire qui vient d'être obtenue dans l'exemple précédent.

- $Cl(S) = S + A + B$ et $Cl(A) = Cl(B) = A + B$.
- De même, la définition de $\mathbf{m}(X)$ nous donne :

$$\mathbf{m}(S) = aAB + aA + aB + BA + a + b$$

$$\mathbf{m}(A) = BBB + BB + a$$

$$\mathbf{m}(B) = AB + b$$

La grammaire G' est donc définie par les règles globales suivantes :

$$S \longrightarrow \mathbf{m}(S) + \mathbf{m}(A) + \mathbf{m}(B)$$

$$A \longrightarrow \mathbf{m}(A) + \mathbf{m}(B)$$

$$B \longrightarrow \mathbf{m}(A) + \mathbf{m}(B)$$

c'est-à-dire :

$$S \longrightarrow aAB + aA + aB + BA + BBB + BB + AB + a + b$$

$$A \longrightarrow BBB + BB + AB + a + b$$

$$B \longrightarrow BBB + BB + AB + a + b$$

- Compte tenu du fait que $\bar{B} = \bar{A}$

$$\bar{V} = \bar{S} + \bar{A}$$

et les règles globales de \bar{G} sont

$$\bar{S} \longrightarrow a\bar{A}\bar{A} + a\bar{A} + \bar{A}\bar{A} + \bar{A}\bar{A}\bar{A} + a + b$$

$$\bar{A} \longrightarrow \bar{A}\bar{A}\bar{A} + \bar{A}\bar{A} + a + b.$$

G. Grammaires propres.

- G est propre ssi G ne produit pas ε et n'admet pas d' ε -dérivation.

Nettoyage des grammaires

Il existe une grammaire propre $G' = (\mathcal{V}', \mathcal{A}, R')$ et une application $c : \mathcal{V} \rightarrow \mathcal{V}'$ qui vérifient $\mathcal{L}(G', c(X)) = \mathcal{L}(G, X) - \varepsilon$ pour toute $X \in \mathcal{V}$.

Il suffit d'appliquer à G successivement la transformation E puis la transformation F : la grammaire obtenue s'appelle la grammaire propre équivalente à G .

Attention.



I faut procéder dans cet ordre, non seulement parce que nous avons supposé que G ne produisait pas ε pour effectuer la transformation F, mais aussi parce que E peut introduire des ε -dérivations!

5 – Annexe.

Nous allons voir des applications où la notion de grammaire propre joue un rôle très important.

5.1 – Lemme d'itération.

Ce lemme (*pumping lemma*) exprime une propriété nécessaire pour qu'un langage soit algébrique : malheureusement, cette propriété n'est pas suffisante pour cela!

Lemme d'itération

Soit L un langage algébrique, alors il existe un entier $N > 0$ tel que si $u \in L$ et $|u| \geq N$, on peut trouver cinq mots u_1, u_2, u_3, v et w satisfaisant les propriétés suivantes :

- 1) $u = u_1vu_2wu_3$;

- 2) $vw \neq \varepsilon$;

- 3) $|vu_2w| \leq N$;

- 4) $u_1v^k u_2 w^k u_3 \in L$ pour tout entier naturel k .

(“pompage”)

Rappelons tout d'abord quelques propriétés élémentaires bien connues des arbres finis :

Soit \mathbf{A} un arbre de hauteur h et dont les embranchements sont d'ordre au plus égal à l :

- le nombre de feuilles de \mathbf{A} est au plus égal à l^h ;
- h est la longueur maximale des branches de \mathbf{A} .

Soient B une branche de \mathbf{A} de longueur h , X un nœud appartenant à B et \mathbf{X} le sous-arbre de racine X :

- la partie de B qui se trouve dans X est de longueur maximale dans X et sa longueur est donc égale à la hauteur de ce dernier.

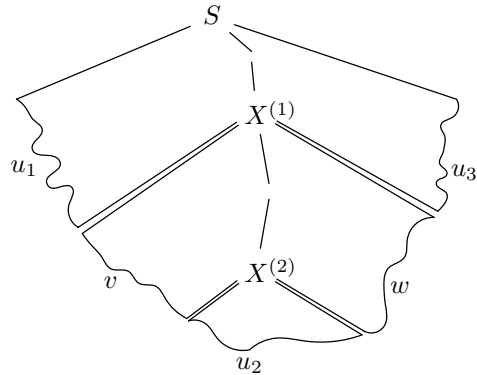
Venons-en à la démonstration elle-même.

Si L est fini, il suffit de prendre pour N un entier strictement supérieur à la longueur maximale des mots de L pour satisfaire le lemme.

Sinon, soient $G = (\mathcal{V}, \mathcal{A}, R)$ une grammaire propre, $S \in \mathcal{V}$ telle que $\mathcal{L}(G, S) = L - \varepsilon$, $V = |\mathcal{V}|$ le nombre de ses variables et l la longueur maximale des α tels que l'on ait $(X \rightarrow \alpha) \in R$.

Nous allons montrer que $N = l^{V+1}$ satisfait le lemme.

Considérons $u \in L$ tel que $|u| \geq N$ et choisissons une dérivation $S \Rightarrow u$: l'arbre \mathbf{A} de cette dérivation admet $|u|$ feuilles, sa hauteur est donc au moins égale à $V + 1$. Soit B une branche de longueur maximale : elle comporte au moins $V + 1$ nœuds étiquetés par des variables, or celles-ci sont au nombre de V , l'une au moins d'entre elles figure donc au moins deux fois dans B : qualifions de "redoublante" toute variable qui possède cette propriété. Soit X une variable redoublante et désignons par $X^{(1)}$ et $X^{(2)}$ deux occurrences distinctes de X figurant dans B , la première choisie plus proche de la racine que la seconde.



En élagant \mathbf{A} d'abord en $X^{(2)}$ puis en $X^{(1)}$ on obtient une dérivation composée

$$S \Rightarrow u_1 X^{(1)} u_3 \xrightarrow{n} u_1 v X^{(2)} w u_3 \Rightarrow u_1 v u_2 w u_3 = u.$$

- La dérivation intermédiaire $d : X \xrightarrow{n} v X w$ est de longueur $n > 0$ et on a donc $vw \neq \varepsilon$ puisque G est propre.
- En substituant à d la dérivation $d^k : X \xrightarrow{kn} v^k X w^k$, définie pour chaque entier k comme la composée de k applications de d , qui se définit par récurrence sur k :

$$\begin{aligned} d^0 : X &\xrightarrow{0} X \\ d^{k+1} : X &\xrightarrow{n} v X w \xrightarrow{kn} v^{k+1} X w^{k+1}, \end{aligned}$$

dans la dérivation précédente, on obtient :

$$S \Rightarrow u_1 X u_3 \xrightarrow{kn} u_1 v^k X w^k u_3 \Rightarrow u_1 v^k u_2 w^k u_3$$

dont l'existence prouve que $u_1 v^k u_2 w^k u_3 \in L$.

- Il nous reste à satisfaire la condition $|v u_2 w| \leq N$: parmi les variables redoublantes, désignons par X celle pour laquelle $X^{(1)}$ est le plus près possible de la feuille de B . A part elle-même, aucune autre variable redoublante ne se trouve dans la partie B' de B qui commence en $X^{(1)}$. La longueur de B' est donc au plus égale à $V + 1$, le nombre des feuilles du sous-arbre de racine $X^{(1)}$, c'est-à-dire $|v u_2 w|$, est donc au plus égal à $l^{V+1} = N$: n'est-ce pas ce que nous voulions ?

Se lemme (et ses nombreux raffinements, dont deux sont proposés en exercices) est un outil très efficace pour montrer qu'un langage n'est pas algébrique ; cependant, il ne peut servir à rien lorsque l'on veut montrer qu'un langage est algébrique car ce lemme est une implication dont **la réciproque est fautive** : il existe effectivement des grammaires qui vérifient les conclusions du lemme sans être algébriques pour cela.

Appliquons le lemme pour vérifier que le langage $L = \{a^m b^m c^m \mid m \geq 0\}$ n'est pas algébrique (a, b et c sont supposées distinctes deux à deux).

Nous allons montrer que pour tout $N > 0$ et tout $u \in L$ tel que $|u| \geq N$, une décomposition satisfaisant 1), 2) et 3) ne peut satisfaire 4).

Prenons $u = a^N b^N c^N$ et considérons une décomposition $u = u_1 v u_2 w u_3$ satisfaisant 2) et 3). Il découle de 3) que $v u_2 w$ ne peut contenir simultanément une occurrence de a et de c , regardons les cas qui restent possibles :

- si $v u_2 w$ ne contient des occurrences que d'une seule lettre, par exemple b , avec $k = 0$ la propriété 4) produit un mot $a^N b^m c^N$, où $m < N$ grâce à 2) : un tel mot n'est pas dans L ;
- sinon
 - si v et w ne comportent chacun que des occurrences d'une seule lettre : on aboutit à une réfutation de 4) du type précédent;
 - sinon, v par exemple est mixte : $v = a^l b^m$ pour $l > 0$ et $m > 0$, alors $v^2 = a^l b^m a^l b^m$ n'est pas un facteur d'un mot de L .

5.2 – Forme normale de Chomsky.

G est une grammaire en *forme normale de Chomsky* ssi ses règles sont de l'une des formes :

$$X \longrightarrow YZ \text{ ou } X \longrightarrow x$$

pour $Y \in \mathcal{V}, Z \in \mathcal{V}$ et $x \in \mathcal{A}$.

Une telle grammaire est propre, de plus, il est évident que l'intérêt des grammaires de Chomsky réside dans la forme particulièrement simple et régulière de leurs règles.

Forme normale de Chomsky

Pour toute grammaire propre $G = (\mathcal{V}, \mathcal{A}, R)$ il existe une grammaire $G' = (\mathcal{V}', \mathcal{A}, R')$ en forme normale de Chomsky telle que

- $\mathcal{V} \subseteq \mathcal{V}'$,
 - pour toute $X \in \mathcal{V}, \mathcal{L}(G', X) = \mathcal{L}(G, X)$.
-

On dit que la grammaire ainsi construite est en *forme normale de Chomsky* de G .

Pour construire la grammaire G' il est nécessaire d'introduire de nouvelles variables :

- pour chaque $\alpha \in (\mathcal{A} + \mathcal{V})^* - \varepsilon$, on introduit une variable notée $[\alpha]$,
- chaque $X \in \mathcal{V}$ est identifiée à $[X]$,
- par contre, $x \in \mathcal{A}$ n'est pas identifiée à $[x]$.

On définit alors une grammaire $G_\alpha = (\mathcal{V}_\alpha, \mathcal{A}, R_\alpha)$, par récurrence sur $\alpha \in (\mathcal{A} + \mathcal{V})^* - \varepsilon$:

- si $X \in \mathcal{V} : \mathcal{V}_X = X$ et $R_X = \emptyset$,
- si $x \in \mathcal{A} : \mathcal{V}_x = [x]$ et $R_x = ([x] \longrightarrow x)$,
- si $|\alpha| > 0$ et si $\xi \in \mathcal{A} + \mathcal{V} : \mathcal{V}_{\alpha\xi} = [\alpha\xi] + \mathcal{V}_\alpha + \mathcal{V}_\xi$ et $R_{\alpha\xi} = ([\alpha\xi] \longrightarrow [\alpha][\xi]) + R_\alpha + R_\xi$.

La grammaire ainsi définie est en forme normale de Chomsky.

On a évidemment $[\alpha] \xrightarrow{*}_{G_\alpha} \alpha$ et réciproquement, si $[\alpha] \xrightarrow{*}_{G_\alpha} \beta$ avec $\beta \in (\mathcal{A} + \mathcal{V})^*$, alors $\beta = \alpha$.

Exemple.

Soit $\alpha = aAbBAa$, alors

$$\mathcal{V}_\alpha = [aAbBAa] + [aAbBA] + [aAbB] + [aAb] + [aA] + [a] + A + [b] + B$$

R_α est l'ensemble des règles

$$\begin{aligned} [aAbBAa] &\longrightarrow [aAbBA][a] \\ [aAbBA] &\longrightarrow [aAbB]A \\ [aAbB] &\longrightarrow [aAb]B \\ [aAb] &\longrightarrow [aA][b] \\ [aA] &\longrightarrow [a]A \\ [a] &\longrightarrow a \end{aligned}$$

$$[b] \longrightarrow b$$

Une dérivation $[aAbBAa] \Longrightarrow aAbBAa$ est alors facile à construire!

Démonstration de la propriété.

Soit $G = (\mathcal{V}, \mathcal{A}, R)$ une grammaire propre.

On construit G' de proche en proche de la façon suivante :

- R' contient initialement les règles de G qui sont de la forme $X \longrightarrow YZ$ (pour $Y \in \mathcal{V}$ et $Z \in \mathcal{V}$) ou de la forme $X \longrightarrow x$ (pour $x \in \mathcal{A}$), et \mathcal{V}' est initialement égal à \mathcal{V} .

- Les autres règles de G ont la forme $(X \longrightarrow \alpha\xi) \in R$ avec $|\alpha| > 1$ (car G est propre) et $\xi \in \mathcal{A} + \mathcal{V}$. Pour chacune d'entre elles :

- on adjoint $\mathcal{V}_\alpha + \mathcal{V}_\xi$ à \mathcal{V}' ,
- on adjoint $(X \longrightarrow [\alpha][\xi]) + R_\alpha + R_\xi$ à R' .

On vérifie que $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}$. Nous ferons simplement une remarque utile pour la preuve de $\mathcal{L}(G', \mathcal{V}) \subseteq \mathcal{L}(G, X)$: toute dérivation $[\alpha] \xrightarrow[k']{G'} u$ avec $u \in \mathcal{A}^*$, peut se transformer en

une dérivation $[\alpha] \xrightarrow[l']{G'} \alpha \xrightarrow[m']{G'} u$ telle que $[\alpha] \xrightarrow[l']{G'} \alpha$ se passe entièrement dans la grammaire G_α . Cette remarque se montre par induction sur α :

- si $\alpha = X \in \mathcal{V}$, on a $[\alpha] = X \xrightarrow{0} X$;
- sinon, si $\alpha = x \in \mathcal{A}$, $[x] \longrightarrow x$ est la seule règle que l'on puisse appliquer;
- sinon, $\alpha = \beta\xi$, $[\beta\xi] \longrightarrow [\beta][\xi]$ est la seule règle que l'on puisse appliquer : il reste à appliquer l'hypothèse d'induction aux deux dérivations que l'on peut obtenir en décomposant $[\beta][\xi] \xrightarrow[n']{G'} u$.

Exemple.

Calculons une forme normale de Chomsky de la grammaire propre sur $\mathcal{V} = S + A + B$ et $\mathcal{A} = a + b$ dont les règles globales sont

$$\begin{aligned} S &\longrightarrow aAB + BBa + BA \\ A &\longrightarrow BBB + aA + a \\ B &\longrightarrow AS + b \end{aligned}$$

On a $\mathcal{V}' = S + A + B + [aA] + [BB] + [a]$ et les règles globales de G' sont

$$\begin{aligned} S &\longrightarrow [aA]B + [BB][a] + BA \\ A &\longrightarrow [BB]B + [a]A + a \\ B &\longrightarrow AS + b \\ [aA] &\longrightarrow [a]A \\ [BB] &\longrightarrow BB \\ [a] &\longrightarrow a \end{aligned}$$

5.3 – Forme normale de Greibach.

Une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ peut être regardée comme un principe général de programmation récurive lié aux langages qu'elle peut engendrer ou reconnaître : la présence d'une variable dans la partie droite d'une règle est alors l'appel à un sous-programme. Dans cet esprit, on pose les définitions suivantes, relatives à G :

Définitions.

- Une variable $X \in \mathcal{V}$ est *réursive à gauche* ssi il existe une dérivation non triviale $X \xrightarrow{k+1} X\alpha$.
- Une règle *fait un appel à gauche* si elle est de la forme $X \rightarrow Y\alpha$ où l'occurrence explicite de $Y \in \mathcal{V}$ constitue l'appel à gauche en question et où $\alpha \in (\mathcal{A} + \mathcal{V})^*$.
- G est une grammaire en *forme normale de Greibach* ssi
 - G ne produit pas ε ;
 - aucune de ses règles ne fait d'appel à gauche.

En particulier, une grammaire en *forme normale de Greibach* est propre et aucune de ses variables n'est réursive à gauche.

Forme normale de Greibach

Pour toute grammaire propre $G = (\mathcal{V}, \mathcal{A}, R)$ il existe une grammaire $G' = (\mathcal{V}', \mathcal{A}, R')$ en forme normale de Greibach telle que :

- $\mathcal{V} \subseteq \mathcal{V}'$;
 - pour toute $X \in \mathcal{V}$, $\mathcal{L}(G', X) = \mathcal{L}(G, X)$.
-

La grammaire ainsi obtenue s'appelle *une forme normale de Greibach de G* .

Il existe deux méthodes classiques pour faire la construction satisfaisant la propriété précédente :

- La première utilise des opérations de *substitution* et de *résolution partielle* adaptées de celles qui ont servi à **la résolution d'un système linéaire par la méthode de Gauss** dans le chapitre 1 : c'est celle que nous étudierons ici. Elle augmente considérablement le nombre des règles.
- La seconde, plus globale, utilise une méthode matricielle : elle augmente considérablement le nombre de variables (cf. exercice 22).

Dans toute la suite G est une grammaire *propre*.

Substitution ou composition à gauche.

Soient A et B deux variables : la règle globale de A peut s'écrire $A \rightarrow B\mathbf{p} + \mathbf{q}$ où les éléments de \mathbf{q} ne commencent pas par une occurrence de B (c'est-à-dire $\mathbf{q} \subseteq (\mathcal{A} + \mathcal{V})^* - B(\mathcal{A} + \mathcal{V})^*$).

Soit $B \rightarrow \mathbf{n}$ la règle globale de B :

La *composition à gauche de A par B* est la substitution partielle qui consiste à remplacer la règle globale $A \rightarrow B\mathbf{p} + \mathbf{q}$ de A par la nouvelle règle globale $A \rightarrow \mathbf{np} + \mathbf{q}$.

Soit $G' = (\mathcal{V}, \mathcal{A}, R')$ la grammaire ainsi obtenue, il est facile de vérifier que $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}$.

Résolution partielle et dérécursion à gauche.

Soit $A \rightarrow A\mathbf{p} + \mathbf{q}$ la règle globale de $A \in \mathcal{V}$, où les éléments de \mathbf{q} ne commencent pas par une occurrence de A (c'est-à-dire $\mathbf{q} \subseteq (\mathcal{A} + \mathcal{V})^* - A(\mathcal{A} + \mathcal{V})^*$) : on se propose de modifier la grammaire de telle sorte que tout appel à gauche disparaisse, sans pour cela modifier les langages qu'elle engendre.

L'application des résultats de la section 3 du présent chapitre, montre que $\mathcal{L}(A) = \mathcal{L}(A)\mathcal{L}(\mathbf{p}) + \mathcal{L}(\mathbf{q})$: en se reportant aux résultats du chapitre 1 sur les équations linéaires (section 4.1) il n'est donc pas invraisemblable que l'on ait $\mathcal{L}(A) = \mathcal{L}(\mathbf{q})\mathcal{L}(\mathbf{p})^* = \mathcal{L}(\mathbf{qp}^*)$.

Si donc, on admettait des règles globales dont le second membre n'est pas nécessairement fini, on pourrait tenter de remplacer la règle de A par $A \rightarrow \mathbf{qp}^*$ (le second membre est seulement régulier) : cette façon de procéder est justifiée dans l'exercice 10.

Pour respecter la définition d'une grammaire, on peut introduire une nouvelle variable \bar{A} telle que $\mathcal{L}(\bar{A}) = \mathcal{L}(\mathbf{p})^*$ et remplacer la règle globale de A par les règles globales suivantes

$$A \rightarrow \mathbf{q}\bar{A}$$

$$\bar{A} \longrightarrow p\bar{A} + \varepsilon$$

L'élimination de la production de ε par \bar{A} conduit à l'opération suivante, qui est généralement adoptée ici.

La *dérécurSION à gauche* de A est l'opération suivante :

– adjoindre une nouvelle variable \bar{A} à \mathcal{V} ;

– remplacer la règle globale $A \longrightarrow Ap + q$ de A par les règles globales :

$$\begin{array}{l} A \longrightarrow q(\bar{A} + \varepsilon) \\ \bar{A} \longrightarrow p(\bar{A} + \varepsilon) \end{array} \quad \begin{array}{l} \text{(c'est-à-dire } A \longrightarrow q\bar{A} + q) \\ \text{(c'est-à-dire } \bar{A} \longrightarrow p\bar{A} + p) \end{array}$$

Remarques.

- Lorsque A ne fait aucun appel récursif à gauche, c'est-à-dire lorsque l'on a $p = \emptyset$, la variable \bar{A} engendre \emptyset : l'introduction de cette variable est donc parfaitement inutile!

- G étant propre, on a $\varepsilon \notin q$ et $\varepsilon \notin p$, ce qui, compte tenu de la définition de q implique que, après la dérécurSION de A :

– aucune règle de A ne fait un appel à gauche par A ou par \bar{A} ,

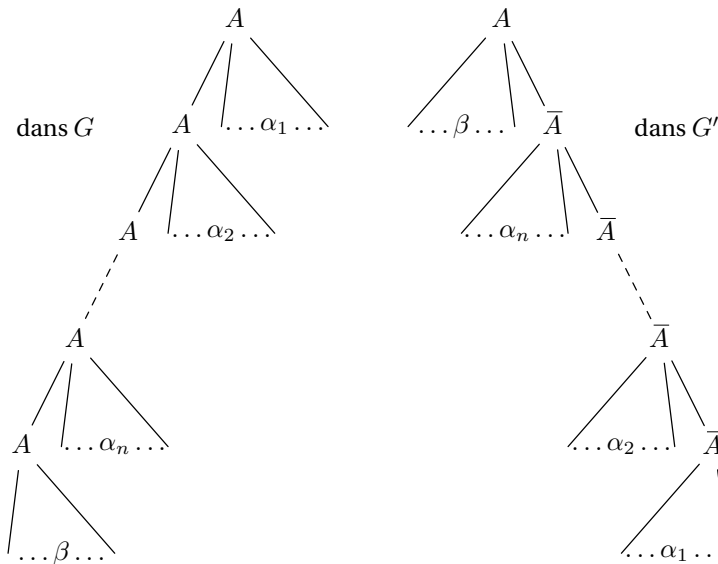
– aucune règle de \bar{A} ne fait un appel à gauche par \bar{A} , mais, comme il est possible qu'un élément de p commence par A , il peut y avoir des règles de \bar{A} qui font un appel à gauche par A : il faudra corriger cet éventuel défaut.

Propriété.

Soit $G' = (\mathcal{V} + \bar{\mathcal{A}}, \mathcal{A}, R')$ la grammaire obtenue en appliquant cette opération, alors $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}$.

La démonstration de cette égalité repose sur la possibilité, illustrée par la figure ci-dessous, de transformer une dérivation $A \xrightarrow[G]{n+1} \gamma$ qui est l'enchaînement de n règles parmi les $A \xrightarrow[G]{p}$ et d'une règle parmi les

$A \xrightarrow[G]{q}$ en une dérivation $A \xrightarrow[G']{n+1} \gamma$, et réciproquement.



On peut conclure en faisant des inductions sur le nombre de tels enchaînements dans les arbres de dérivation.

Nous sommes en mesure de calculer une forme normale de Greibach de G : en s'inspirant du vocabulaire utilisé pour les systèmes linéaires, nous allons mettre un système sous forme "sous-triangulaire" en appliquant les deux opérations qui viennent d'être décrites.

Pour fixer les idées, nous allons écrire des procédures qui montrent comment ces applications peuvent être organisées.

Pour cela, il est commode d'énumérer les variables $\mathcal{V} = A_1 + \dots + A_N$.

A. Considérons la procédure itérative suivante :

```

pour i croissant de 1 à N faire
    pour j croissant de 1 à i - 1 faire
        composer Ai à gauche par Aj
    fin
dérécursifier Ai à gauche
fin

```

Après l'exécution de cette procédure :

- une variable A_i ne fait éventuellement des appels à gauche que par des variables A_j pour lesquelles $j > i$, en particulier, A_N ne fait aucun appel à gauche;
- une variable \bar{A}_i ne fait éventuellement des appels à gauche que par des variables $A_j \in \mathcal{V}$.

Exemple.

Considérons la grammaire propre sur $\mathcal{V} = A_1 + A_2 + A_3$ et $\mathcal{A} = a + b$ dont les règles globales sont

$$\begin{aligned} A_1 &\longrightarrow A_2 A_3 + a \\ A_2 &\longrightarrow A_3 A_1 + A_1 b \\ A_3 &\longrightarrow A_1 A_2 + A_2 A_1 + b \end{aligned}$$

L'exécution du processus itératif précédent se déroule ainsi :

$i = 1$ la petite boucle ne fait rien

- A_1 n'est pas récursive à gauche;

$i = 2$: $j = 1$

$$A_2 \longrightarrow A_3 A_1 + (A_2 A_3 + a)b$$

c'est-à-dire :

$$A_2 \longrightarrow A_2 A_3 b + A_3 A_1 + ab$$

- dérécursion à gauche de A_2 :

$$\bar{A}_2 \longrightarrow (A_3 A_1 + ab)(\bar{A}_2 + \varepsilon)$$

$$\bar{A}_2 \longrightarrow A_3 b(\bar{A}_2 + \varepsilon)$$

$i = 3$: $j = 1$

$$A_3 \longrightarrow (A_2 A_3 + a)A_2 + A_2 A_1 + b$$

c'est-à-dire :

$$A_3 \longrightarrow A_2(A_3 A_2 + A_1) + aA_2 + b$$

$i = 3$: $j = 2$

$$A_3 \longrightarrow (A_3 A_1 + ab)(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1) + aA_2 + a$$

c'est-à-dire :

$$A_3 \longrightarrow A_3 A_1(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1) + ab(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1) + aA_2 + b$$

- dérécursion à gauche de A_3 :

$$\bar{A}_3 \longrightarrow (ab(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon)$$

$$\bar{A}_3 \longrightarrow A_1(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1)(\bar{A}_3 + \varepsilon)$$

Ceci donne les règles globales suivantes :

$$A_1 \longrightarrow A_2 A_3 + a$$

$$\bar{A}_2 \longrightarrow (A_3 A_1 + ab)(\bar{A}_2 + \varepsilon)$$

$$\bar{A}_2 \longrightarrow A_3 b(\bar{A}_2 + \varepsilon)$$

$$\bar{A}_3 \longrightarrow (ab(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon)$$

$$\bar{A}_3 \longrightarrow A_1(\bar{A}_2 + \varepsilon)(A_3 A_2 + A_1)(\bar{A}_3 + \varepsilon)$$

B. On peut maintenant éliminer tous les appels à gauche des A_i par des compositions, en exécutant par exemple la boucle :

pour i décroissant de N à 2 faire
 composer toutes les A_j à gauche par A_i
 fin

Après l'exécution de cette procédure, les variables A_i ne font plus aucun appel à gauche.

Exemple (suite).

L'exécution de cette procédure produit les règles globales suivantes :

$$\begin{aligned} A_3 &\longrightarrow (ab(\bar{A}_2 + \varepsilon)(A_3A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon) \\ A_2 &\longrightarrow ((ab(\bar{A}_2 + \varepsilon)(A_3A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon)A_1 + ab)(\bar{A}_2 + \varepsilon) \\ A_1 &\longrightarrow (((ab(\bar{A}_2 + \varepsilon)(A_3A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon)A_1 + ab)(\bar{A}_2 + \varepsilon)A_3 + a \end{aligned}$$

C. Il reste à éliminer les appels à gauche des nouvelles variables : ceux-ci se font tous par des variables d'origine, il suffit donc de faire des compositions à gauche.

Exemple (suite).

Par les compositions appropriées, les règles globales de \bar{A}_2 et \bar{A}_3 deviennent :

$$\begin{aligned} \bar{A}_2 &\longrightarrow (ab(\bar{A}_2 + \varepsilon)(A_3A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon)b(\bar{A}_2 + \varepsilon) \\ \bar{A}_3 &\longrightarrow (((ab(\bar{A}_2 + \varepsilon)(A_3A_2 + A_1) + aA_2 + b)(\bar{A}_3 + \varepsilon)A_1 + ab)(\bar{A}_2 + \varepsilon)A_3 + a) \\ &\quad (\bar{A}_2 + \varepsilon)(A_3A_2 + A_1)(\bar{A}_3 + \varepsilon) \end{aligned}$$

Il resterait à faire quelques observations simples pour vérifier que la grammaire est bien une forme normale de Greibach de G .

Comme promis, le nombre des variables reste raisonnable (pas plus que $2N$), mais celui des règles est maintenant extrêmement élevé : 27 pour A_1 , 26 pour A_2 , 12 pour A_3 , 24 pour \bar{A}_2 et 216 pour \bar{A}_3 !

EXERCICES.

Lorsqu'aucune autre précision n'est donnée, \mathcal{A} est un alphabet fini quelconque et G désigne une grammaire $(\mathcal{V}, \mathcal{A}, R)$.

Exercice 1.

Montrer que les langages suivants sont algébriques (a et b sont distincts l'un de l'autre) :

$$L_1 = \{a^m b^m \mid m \geq 0\}$$

$$L_2 = \{a^m b^n \mid 0 \leq m < n\}$$

$$L_3 = \{a^m b^n a^n \mid m \geq 0, n \geq 0\}$$

$$L_4 = \{uc\tilde{u} \mid u \in \mathcal{A}^*\}$$

$$L_5 = \{uv \mid u \in (a+b)^*, v \in (a+b)^*, |u| = |v|, v \neq \tilde{u}\}$$

Exercice 2. Les palindromes et les autres.

L'ensemble $Pal \subseteq \mathcal{A}^*$ des *palindromes* sur l'alphabet \mathcal{A} (abréviation de $Pal(\mathcal{A})$) est défini par

$$u \in Pal \text{ ssi } \tilde{u} = u$$

(L'opération *image miroir* $u \mapsto \tilde{u}$ qui renverse l'ordre des caractères de u se définit par récurrence de la façon suivante :

$$\tilde{\varepsilon} = \varepsilon \quad \text{et} \quad \tilde{ux} = x\tilde{u} \text{ pour tout } u \in \mathcal{A}^* \text{ et tout } x \in \mathcal{A},$$

où l'on a utilisé l'adjonction à droite et à gauche).

a) Montrer que $u \in Pal$ ssi

ou bien il existe $v \in \mathcal{A}^*$ tel que $u = v\tilde{v}$

ou bien il existe $v \in \mathcal{A}^*$ et $x \in \mathcal{A}$ tels que $u = vx\tilde{v}$.

Utiliser ce résultat pour construire une grammaire qui engendre le langage Pal dans le cas où $\mathcal{A} = a + b$.

b) Soit $\overline{Pal} \subseteq \mathcal{A}^*$ l'ensemble des mots sur \mathcal{A} qui **ne sont pas des palindromes**.

Montrer que $u \in \overline{Pal}$ ssi il existe $v \in \mathcal{A}^*$, $x, y \in \mathcal{A}$ et $w \in \mathcal{A}^*$ tels que

$$u = vxw\tilde{v} \quad \text{et} \quad x \neq y.$$

Utiliser ce résultat pour construire une grammaire qui engendre le langage \overline{Pal} dans le cas où $\mathcal{A} = a + b$.

Exercice 3.

Montrer que si $L \subseteq \mathcal{A}^*$ est algébrique alors $\tilde{L} = \{u \in \mathcal{A}^* \mid \tilde{u} \in L\}$ l'est aussi.

Exercice 4.

Soit $\mathcal{A} = a + b$, où a et b sont distincts l'un de l'autre.

On considère le langage $L \subseteq \mathcal{A}^*$ défini par

$$u \in L \text{ ssi } |u|_a = |u|_b + 1 \text{ et } |v|_a \leq |v|_b \text{ pour tout } v \in fgs(u)$$

quel que soit $u \in \mathcal{A}^*$,

et la grammaire G définie par la seule règle globale $S \longrightarrow bSS + a$.

Montrer que $L = \mathcal{L}(G, S)$.

Indication. Utiliser le résultat de l'exercice 1.9.

Exercice 5. Autant de a que de b dans le même mot.

Soit $L \subseteq \mathcal{A}^*$ le langage constitué des mots ayant autant d'occurrences de a que de b , c'est-à-dire :

$$u \in L \text{ ssi } |u|_a = |u|_b.$$

Soit G la grammaire sur \mathcal{A} comportant une seule variable S et dont la règle globale est :

$$S \longrightarrow SS + aSb + bSa + \varepsilon$$

Montrez que $L = \mathcal{L}(G, S)$.

Indication. Utiliser le résultat de l'exercice 1.18.

Exercice 6. Deux fois plus de a que de b dans le même mot.

Soient $\mathcal{A} = a + b$ un alphabet de deux lettres et $G = (S, \mathcal{A}, R)$ la grammaire pour laquelle R est défini par la règle globale

$$S \longrightarrow \varepsilon + SaSaSbS + SaSbSaS + SbSaSaS.$$

Le but de cet exercice est de montrer que $\mathcal{L}(G, S)$ est l'ensemble L des $u \in \mathcal{A}^*$ vérifiant $|u|_a = 2|u|_b$.

a) Montrer que $\mathcal{L}(G, S) \subseteq L$ en effectuant une induction sur la longueur des dérivations.

b) Montrer que si $u \in L - \varepsilon$ alors u a un facteur $xyz \in L$ de longueur 3.

Indication. C'est le résultat de l'exercice 1.19 pour $\alpha = 1$ et $\beta = 2$. (Une preuve "directe" est possible mais, assez périlleuse et difficilement généralisable.)

c) Pour tout $u \in \mathcal{A}^*$ on définit $\bar{u} \in (\mathcal{A} + S)^*$, de la façon suivante :

- $\bar{\varepsilon} = \varepsilon$,
- $\bar{x} = x$ pour tout $x \in \mathcal{A}$,
- $\overline{ux} = \bar{u}Sx$ pour tout $u \in \mathcal{A}^* - \varepsilon$, et pour tout $x \in \mathcal{A}$.

\bar{u} est donc obtenu en intercallant S entre les lettres de u , par exemple $\bar{u} = aSaSb$ pour $u = aab$.

Vérifier que $\overline{uv} = \bar{u}S\bar{v}$ lorsque $u \neq \varepsilon$ et $v \neq \varepsilon$.

d) Montrer les deux propriétés suivantes :

- $\bar{u} \xrightarrow{*} u$ pour tout $u \in \mathcal{A}^*$.
- $S \xrightarrow{*} S\bar{u}S$ pour tout $u \in L - \varepsilon$ (utiliser **b** pour faire une induction).

e) En déduire que $L \subseteq \mathcal{L}(G, S)$.

f) Utiliser les idées et les résultats de cet exercice et l'exercice 1.19 pour montrer que le langage $L \subseteq \mathcal{A}^*$ défini par

$$u \in L \text{ ssi } \alpha|u|_a = \beta|u|_b$$

(où α et β sont des entiers naturels non nuls) est algébrique.

(On trouvera, en particulier, une nouvelle grammaire engendrant le langage étudié dans l'exercice précédent.)

Exercice 7. Les opérations régulières.

a) Montrer que si L_1 et L_2 sont algébriques, alors $L_1 + L_2$ et L_1L_2 le sont aussi.

b) Montrer que si L est algébrique, alors L^* l'est aussi.

La classe des langages algébriques est donc stable par les opérations régulières.

c) En déduire que les langages réguliers sont algébriques.

Remarque. Ce résultat a déjà été obtenu par la considération des grammaires linéaires à droite. La réciproque n'est pas vraie, par exemple : $L = \{a^m b^m \mid m \geq 0\}$ est algébrique mais **il n'est pas régulier**.

d) Appliquer **a)** et **b)** pour construire une grammaire engendrant le langage $a^2(a + ab + ba)^*$ à partir de l'une de ses variables.

Exercice 8. Grammaires linéaires à droite.

Les grammaires linéaires à droite que nous avons utilisées dans la section 1.3 étaient très particulières. En général, on appelle ainsi une grammaire dont chaque règle est de l'une des formes suivantes :

$$X \longrightarrow uY \quad X \longrightarrow u$$

où X et Y sont des variables et $u \in \mathcal{A}^*$.

Montrer que pour toute grammaire G linéaire à droite en ce sens, il existe une grammaire $G' = (\mathcal{V}', \mathcal{A}, R')$ linéaire à droite au sens de la section 1.3, telle que $\mathcal{V} \subseteq \mathcal{V}'$ et $\mathcal{L}(G', X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}$.

Exercice 9. Image d'un langage algébrique par une substitution.

Soient \mathcal{A} et \mathcal{B} deux alphabets finis et $f : \mathcal{A} \rightarrow \mathcal{P}(\mathcal{B}^*)$ une substitution telle que $f(x)$ est algébrique pour chaque $x \in \mathcal{A}$.

Montrer que si $L \subseteq \mathcal{A}^*$ est algébrique alors $f(L) \subseteq \mathcal{B}^*$ l'est aussi.

En particulier, vérifier que si L est algébrique, alors le langage $sm(L)$ de ses sous-mots l'est aussi (cf. exercice 1.17).

Exercice 10. Image inverse d'un langage algébrique par un homomorphisme

Soit $f : \mathcal{A} \rightarrow \mathcal{B}^*$ un homomorphisme : son application inverse $f^{-1} : \mathcal{B}^* \rightarrow \mathcal{P}(\mathcal{A}^*)$ a été considérée dans l'exercice 1.14.

- Montrer que si $M \subseteq \mathcal{B}^*$ est algébrique alors $f^{-1}(M) \subseteq \mathcal{A}^*$ l'est aussi.
- Appliquer ce résultat et l'exercice précédent pour montrer que $tc(L)$ et $usd(L)$ sont algébriques lorsque L l'est (cf. exercice 1.15)

Exercice 11. Les facteurs.

a) Montrer que si L est algébrique alors $Fg(L)$, $Fd(L)$ et $Fact(L)$ le sont aussi (cf. exercice 1.17).

b) Montrer que si L est algébrique alors $fg(L)$, $fd(L)$ et $fact(L)$ le sont aussi.

Indication pour a). Soit $G = (\mathcal{V}, \mathcal{A}, R)$ une grammaire et soit $S \in \mathcal{V}$ telle que $L = \mathcal{L}(G, S)$: nous allons donner les éléments pour construire une grammaire $H = (\mathcal{W}, \mathcal{A} + \langle + \rangle, \Sigma)$ capable d'engendrer $Fact(L) = mel(L, \langle \rangle)$ à partir de l'une de ses variables.

Soit $Q = 0 + 1 + 2$ un alphabet disjoint de \mathcal{A} et \mathcal{V} (l'ensemble des états d'un AFD évident qui reconnaît le langage constitué du seul mot $\langle \rangle$ et qui sera évoqué dans les commentaires d'une définition ci-dessous).

- L'alphabet \mathcal{W} est l'ensemble des symboles $[qXr]$ tels que $(q, X, r) \in Q \times \mathcal{V} \times Q$ vérifie $q \leq r$.

- Pour tout q et tout $r \in Q$, on pose :

$$- [qr] = \begin{cases} \langle & \text{si } q = 0 \text{ et } r = 1; & (\text{comparer à } 0 \bullet \langle = 1) \\ \rangle & \text{si } q = 1 \text{ et } r = 2; & (\text{comparer à } 1 \bullet \rangle = 2) \\ \varepsilon & \text{si } r = q; & (\text{comparer à } q \bullet \varepsilon = q) \\ \emptyset & \text{dans les autres cas.} \end{cases}$$

- Pour tout q et tout $r \in Q$, et tout $x \in \mathcal{A}$, on pose :

$$- [qxr] = \begin{cases} x & \text{si } r = q; \\ \emptyset & \text{sinon.} \end{cases}$$

- Enfin, pour tout $\alpha \in (\mathcal{A} + \mathcal{V})^*$, tout q et tout $r \in Q$, et tout $\eta \in \mathcal{A} + \mathcal{V} + \varepsilon$, on pose :

$$- [q\alpha\eta r] = \sum_{s \in Q} [q\alpha s][s\eta r].$$

Vérifier que $[q\alpha r] \subseteq (\mathcal{A} + \langle + \rangle + \mathcal{W})^*$ est ainsi toujours bien défini et que c'est **un langage fini**.

Par exemple, on constate que pour tout $u \in \mathcal{A}^*$, on a $[0u2] = Fact(u)$, $[1u2] = Fg(u)$, $[0u1] = Fd(u)$ et $[quq] = u$.

• Si $X \xrightarrow{G} \mathbf{1}(X)$ est la règle globale de X dans G , alors $[qXr] \xrightarrow{H} [q\mathbf{1}(X)r]$ est la règle globale de $[qXr]$ dans H (on a étendu la définition précédente aux langages!).

Note. On peut simplifier H sensiblement en observant qu'il est possible, sans inconvénient, d'identifier toute $[qXq]$ à X elle-même. On constate alors que la restriction à \mathcal{V} de la grammaire ainsi obtenue est égale à G et, plus précisément, que toute règle globale de G est aussi une règle globale de cette grammaire.

- Il reste à démontrer que $\mathcal{L}(H, [0S2]) = Fact(L)$.

De même, on peut démontrer que $\mathcal{L}(H, [1S2]) = Fg(L)$, $\mathcal{L}(H, [0S1]) = Fd(L)$ et, bien entendu $\mathcal{L}(H, [qSq]) = L$.

Exercice 12. Conjugué d'un langage algébrique.

Le but de cet exercice est de démontrer que le langage $Conj(L)$ est algébrique lorsque L l'est (voir exercice 1.4 pour la définition de $Conj$).

Soient $G = (\mathcal{V}, \mathcal{A}, R)$ une grammaire, $S \in \mathcal{V}$ et $L = \mathcal{L}(G, S)$: il s'agit de construire une grammaire $G' = (\mathcal{V}', \mathcal{A}, R')$ et de choisir $T \in \mathcal{V}'$ de telle façon que $\mathcal{L}(G', T) = Conj(L)$.

Pour cela, on considère $\bar{\mathcal{V}} = \{\bar{X} \mid X \in \mathcal{V}\}$ une copie de \mathcal{V} disjointe de \mathcal{V} , $T \notin \mathcal{V} + \bar{\mathcal{V}}$ une variable distincte des précédentes et on pose $\mathcal{V}' = T + \mathcal{V} + \bar{\mathcal{V}}$.

Les règles de G' sont celles de G auxquelles on adjoint les suivantes :

- $T \longrightarrow S$;
- $\bar{S} \longrightarrow \varepsilon$;
- pour toute X et toute $Y \in \mathcal{V}$, pour tout α et tout $\beta \in (\mathcal{A} + \mathcal{V})^*$:
si $X \longrightarrow \alpha Y \beta$ est une règle de G alors $\bar{Y} \longrightarrow \beta \bar{X} \alpha$ est une règle de G'

($Y \beta X \alpha$ s'obtient en appliquant une permutation circulaire aux caractères de $X \alpha Y \beta$);

- pour toute $X \in \mathcal{V}$, tout $x \in \mathcal{A}$, pour tout α et tout $\beta \in (\mathcal{A} + \mathcal{V})^*$:
si $X \longrightarrow \alpha x \beta$ est une règle de G alors $T \longrightarrow \beta \bar{X} \alpha x$ est une règle de G'

($\beta X \alpha x$ s'obtient en appliquant une permutation circulaire aux caractères de $X \alpha x \beta$).

a) Montrer (par induction sur les dérivations) que pour toute X et toute $Y \in \mathcal{V}$, pour tout α et tout $\beta \in (\mathcal{A} + \mathcal{V})^*$:

$$X \xrightarrow{*}_G \alpha Y \beta \quad \text{ssi} \quad \bar{Y} \xrightarrow{*}_{G'} \beta \bar{X} \alpha.$$

b) Utiliser ce résultat pour démontrer que pour toute $X \in \mathcal{V}$, tout $x \in \mathcal{A}$, pour tout α et tout $\beta \in (\mathcal{A} + \mathcal{V})^*$:

$$X \xrightarrow{*}_G \alpha x \beta \quad \text{ssi} \quad T \xrightarrow{*}_{G'} \beta \bar{X} \alpha x.$$

c) Montrer que l'on a bien $\mathcal{L}(G', T) = Conj(L)$.

Exercice 13.

Mettre la grammaire $G = (A + B + C + D + E, a + b, R)$ dont les règles globales sont :

$$\begin{aligned} A &\longrightarrow aAb + a \\ B &\longrightarrow Ab + bCC \\ C &\longrightarrow DD + ba \\ D &\longrightarrow aDB \\ E &\longrightarrow bC \end{aligned}$$

sous une forme réduite successivement pour chacune de ses variables.

Exercice 14.

Mettre la grammaire $G = (A + B + C + D + E + F, a + b, R)$ dont les règles globales sont :

$$\begin{aligned} A &\longrightarrow BC + DE + F \\ B &\longrightarrow BB + E \\ C &\longrightarrow aC + D \\ D &\longrightarrow C + EFaD \\ E &\longrightarrow aE + \varepsilon \\ F &\longrightarrow E + b \end{aligned}$$

sous une forme propre.

Exercice 15.

Soit $X \in \mathcal{V}$ telle que $\mathcal{L}(G, X)$ soit **fini**.

a) Construire une grammaire $G' = (\mathcal{V} - X, \mathcal{A}, R')$ telle que pour toute $Y \in \mathcal{V} - X$ on ait $\mathcal{L}(G', Y) = \mathcal{L}(G, Y)$.

b) En déduire que si L est un langage algébrique infini alors il existe une grammaire G telle que $\mathcal{L}(G, X)$ est infini pour toute $X \in \mathcal{V}$.

Exercice 16.

Soit $L \subseteq \mathcal{A}^*$ un langage sur \mathcal{A} . Une nouvelle lettre $c \notin \mathcal{A}$ étant donnée, on définit $M \subseteq \mathcal{A}^*c\mathcal{A}^*$ (chaque élément de M comporte exactement une occurrence de c) par

$$ucv \in M \text{ ssi } u \in L \text{ tel que } v = \tilde{u}$$

où \tilde{u} désigne l'image miroir de u .

a) Montrer que si L est régulier alors M est algébrique.

On se propose maintenant de démontrer la réciproque de cette propriété.

Lorsque M est fini, L l'est aussi et donc est régulier, sinon, soient $G = (\mathcal{V}, \mathcal{A} + c, R)$ une grammaire et $S \in \mathcal{V}$ telles que $M = \mathcal{L}(G, S)$. On suppose que

(I) G est réduite pour S ,

(II) pour toute $X \in \mathcal{V}$, le langage $\mathcal{L}(G, X)$ est infini,

(on sait, grâce à la propriété de réductibilité des grammaires et l'exercice 14, que ces hypothèses ne sont pas restrictives);

de plus, il est utile de considérer $\mathcal{W} \subseteq \mathcal{V}$ défini par

$$X \in \mathcal{W} \text{ ssi il existe } \alpha, \beta \in (\mathcal{A} + \mathcal{V})^* \text{ tels que } X \xrightarrow{*} \alpha c \beta$$

et de poser $\mathcal{V}' = \mathcal{V} - \mathcal{W}$.

b) Supposons qu'il existe λ et $\mu \in (\mathcal{A} + c + \mathcal{V})^*$ et $\xi \in c + \mathcal{W}$ tels que $S \xrightarrow{*} \lambda \xi \mu$.

1) Utiliser (I) pour montrer que $\lambda \in (\mathcal{A} + \mathcal{V}')^*$ (un argument symétrique montrerait que l'on a aussi $\mu \in (\mathcal{A} + \mathcal{V}')^*$).

2) Utiliser (II) pour montrer qu'en fait on a $\lambda \in \mathcal{A}^*$ (un argument symétrique montrerait que l'on a aussi $\mu \in \mathcal{A}^*$).

c) En déduire que toute règle de G est de la forme $X \rightarrow u\xi v$ où $u, v \in \mathcal{A}^*$ et $\xi \in c + \mathcal{W}$, et en particulier que $\mathcal{W} = \mathcal{V}$.

d) Construire une grammaire linéaire à droite $H = (\mathcal{V}, \mathcal{A}, \Sigma)$ telle que $\mathcal{L}(H, S) = L$.

Exercice 17.

L'exercice précédent se généralise facilement et utilement, de la façon suivante.

Soit $c \notin \mathcal{A}$ une nouvelle lettre. Soient $L \subseteq \mathcal{A}^*$ un langage et $f : \mathcal{A}^* \rightarrow \mathcal{P}(\mathcal{A}^*)$ une application telle que $f(v)$ est fini pour tout $v \in \mathcal{A}^*$.

Considérons maintenant $M \subseteq \mathcal{A}^*c\mathcal{A}^*$ (chaque élément de M comporte exactement une occurrence de c) le langage défini par

$$ucv \in M \text{ ssi } v \in L \text{ et } u \in f(v)$$

pour tout $u, v \in \mathcal{A}^*$.

a) Montrer que si L est régulier alors M est algébrique.

b) Réciproquement, montrer que si M est algébrique alors L est régulier.

Indication. La méthode de l'exercice précédent fait encore merveille.

Remarque. L'hypothèse sur f est vérifiée par les applications de ce type qui ont été considérées dans le premier chapitre (*Conj*, *fg*, ...) et par les substitutions finies (celles qui envoient chaque caractère sur un ensemble fini de mots).

Exercice 18.

$X \in \mathcal{V}$ est dite *réursive* ssi il existe $\alpha \in (\mathcal{A} + \mathcal{V})^*$ et $\beta \in (\mathcal{A} + \mathcal{V})^*$ tels que $\alpha\beta \neq \varepsilon$ et $X \xrightarrow[G]{*} \alpha X \beta$.

a) Montrer que si G est propre et si toutes ses variables sont productives, alors les deux propriétés suivantes sont vraies pour toute $X \in \mathcal{V}$:

- 1) si X est réursive, alors $\mathcal{L}(G, X)$ est infini.
- 2) $\mathcal{L}(G, X)$ est infini ssi il existe $Y \in \mathcal{V}$ qui est réursive et accessible à partir de X .

b) Dédire de 2) ci-dessus que si L est un langage algébrique infini, il existe une grammaire G et $X \in \mathcal{V}$ telles que

- $\mathcal{L}(G, X) = L - \varepsilon$,
- toute $Y \in \mathcal{V}$ est réursive.

Exercice 19.

Soit $G = (S, a + b, R)$ la grammaire pour laquelle R est constitué de la règle globale $S \longrightarrow aSS + b$.

Montrer que $\mathcal{L}(G, S)$ est le plus petit $X \subseteq \mathcal{A}^*$ satisfaisant $X = aXX + b$.

Opérations d'un langage régulier sur un langage algébrique.

L'objet des trois exercices qui suivent est de montrer que si L est un langage algébrique et M un langage régulier alors $L \cap M$, $M^{-1}L$ et $mel(L, M)$ sont des langages algébriques.

Il faut marier la carpe et le lapin, c'est-à-dire une grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ et un AF $\mathbf{A} = (Q, \mathcal{A}, \bullet, I, F)$ (pour éviter toute incompatibilité supplémentaire, on supposera que $Q \cap (\mathcal{A} + \mathcal{V}) = \emptyset$), pour obtenir une grammaire $H = (\mathcal{W}, \mathcal{A}, \Sigma)$ capable d'engendrer le langage algébrique en question.

Dans les trois exemples, on utilise l'alphabet de variables $\mathcal{W} = Q \times \mathcal{V} \times Q$ et on désignera le symbole $(q, X, r) \in \mathcal{W}$ par $[qXr]$.

Exercice 20. Intersection avec un langage régulier.

On veut montrer que $L \cap M$ est algébrique lorsque L l'est et M est régulier.

Dans ce cas, on peut considérer un AFDC $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$. On construit la grammaire $H = (\mathcal{W}, \mathcal{A}, \Sigma)$ de la façon suivante :

La notation $[qXr]$ s'étend de la façon suivante : pour tout q et $r \in Q$ et pour tout $\alpha \in (\mathcal{A} + \mathcal{V})^*$, on définit l'ensemble fini $[q\alpha r] \subseteq (\mathcal{A} + \mathcal{W})^*$ en posant tout d'abord, pour tout $x \in \mathcal{A}$,

$$- [qxr] = \begin{cases} x & \text{si } q \bullet x = r \\ \emptyset & \text{sinon} \end{cases}$$

puis, la récurrence :

$$- [qr] = \begin{cases} \varepsilon & \text{si } q = r \\ \emptyset & \text{sinon} \end{cases}$$

$$- [q\alpha\xi r] = \sum_{s \in Q} [q\alpha s][s\xi r] \text{ pour tout } \alpha \in (\mathcal{A} + \mathcal{V})^* \text{ et tout } \xi \in \mathcal{A} + \mathcal{V}.$$

• si $X \xrightarrow[G]{*} I(X)$ est la règle globale de X dans G , alors $[qXr] \xrightarrow[H]{*} [qI(X)r]$ est la règle globale de $[qXr] \in \mathcal{W}$ dans H (on a étendu la définition précédente aux langages!).

a) Appliquer cette construction à un exemple simple.

b) Montrer que pour tout $[qXr] \in \mathcal{W}$ et tout $u \in \mathcal{A}^*$: $[qXr] \xrightarrow[H]{*} u$ ssi $X \xrightarrow[G]{*} u$ et $q \bullet u = r$.

c) En déduire que l'intersection d'un langage algébrique et d'un langage régulier est algébrique.

Remarque. Ceci implique que $L - M$ est algébrique lorsque L est algébrique et M régulier, car le complémentaire d'un langage régulier est régulier (cf. la section 6.1 du chapitre 2).

Attention. L'intersection de deux langages algébriques n'est pas nécessairement algébrique. Par exemple $L = \{a^m b^m a^n \mid m \geq 0, n \geq 0\}$ et $M = \{a^m b^n a^n \mid m \geq 0, n \geq 0\}$ sont algébriques (cf. exercice 1) mais on verra que $L \cap M = \{a^m b^m a^m \mid m \geq 0\}$ n'est pas algébrique (cf. exercice 23).

Exercice 21. Quotient par un langage régulier.

Montrer que si L est algébrique et M régulier alors $M^{-1}L$ est algébrique (cf. exercice 1.12 et la section 6.2.2 du chapitre 2).

Indication. Considérer, non pas un AFDC $\mathbf{A} = (Q, \mathcal{A}, \bullet, q_0, F)$ reconnaissant M , mais l'AF $\mathbf{B} = (Q + \varrho, \mathcal{A}, \bullet, q_0, F + \varrho)$ suivant, construit à partir de \mathbf{A} , reconnaissant $M\mathcal{A}^*$:

- $\varrho \notin Q$ est un nouvel état et une nouvelle sortie;
- l'action \bullet est étendue à ce nouvel état par :
 - $\varrho \in f \bullet x$ pour tout $x \in \mathcal{A}$ et tout $f \in F$;
 - $\varrho \bullet x = \varrho$ pour tout $x \in \mathcal{A}$.

Tenir compte de ϱ dans la définition de \mathcal{W} et poser

$$- [qxr] = \begin{cases} \varepsilon & \text{si } r \neq \varrho \text{ et } r = q \bullet x; \\ x & \text{si } r = \varrho \text{ et } \varrho \in q \bullet x; \\ \emptyset & \text{dans les autres cas.} \end{cases}$$

Exercice 22. Mélange avec un langage régulier.

Montrer que si L est algébrique et M régulier alors $mel(L, M)$ est algébrique (cf. exercices 1.16 et 1.17).

Indication. On pourra trouver une source d'inspiration dans l'exercice 10 où des cas particuliers de cette question sont traités.

Applications du lemme d'itération pour les langages algébriques (section 5.1).**Exercice 23. Langages algébriques sur un alphabet à une seule lettre.**

Soient a une lettre et $L \subseteq a^*$ un langage algébrique.

Désignons par N un entier dont le lemme d'itération assure l'existence à propos de L et posons $n = N!$ (tout entier i tel que $0 < i \leq N$ divise n).

- a) Dédurre du lemme d'itération que, pour tout $u \in L$, $|u| \geq N$ implique $u(a^n)^* \subseteq L$.
- b) On pose $L_i = L \cap a^{N+i}(a^n)^*$ pour tout entier i tel que $0 < i \leq N$.
Montrer que si $L_i \neq \emptyset$ alors il existe $u_i \in \mathcal{A}^*$ tel que $L_i = u_i(a^n)^*$.
- c) En déduire que L est régulier.

Exercice 24.

Montrer que les langages

$$L_1 = \{a^m b^m a^m \mid m \geq 0\}$$

$$L_2 = \{a^m b^n a^m b^n \mid m \geq 0 \text{ et } n \geq 0\}$$

où a et b sont distinctes entre elles, ne sont pas algébriques.

Indication. Montrer que L_i ne satisfait pas la conclusion du lemme d'itération.

Raffinements du lemme d'itération pour les langages algébriques.

Si f est une application transformant tout langage algébrique en un langage algébrique, alors la propriété :

Pour tout langage algébrique L le langage $f(L)$ satisfait la conclusion du lemme d'itération.

est une variante dudit lemme qui, lorsque f est bien adaptée à L , peut se montrer plus fine que la version initiale.

Exercice 25. Lemme d'itération sur les sous-mots (Ogden).

Nous allons exploiter l'idée précédente en utilisant l'application "sous-mot" sm qui a été étudiée dans l'exercice 1.17 : pour garder un contact raisonnable avec le langage initial, nous agirons avec préméditation. Rappelons que $\mathcal{B} = \mathcal{A} + \overline{\mathcal{A}}$ contient chaque élément $x \in \mathcal{A}$ et sa version marquée $\overline{x} \in \overline{\mathcal{A}}$ et, $m : \mathcal{A} \rightarrow \mathcal{P}(\mathcal{B}^*)$ est la substitution de marquage définie par $m(x) = x + \overline{x}$. Il est commode d'introduire ici une variante $\overline{h} : \mathcal{B} \rightarrow \overline{\mathcal{A}}^*$ de h définie par $\overline{h}(x) = \varepsilon$ et $\overline{h}(\overline{x}) = \overline{x}$ pour tout $x \in \mathcal{A}$.

Le nombre d'occurrences marquées de $u \in \mathcal{B}^*$ sera noté $|u|_m$ et on a évidemment $|u|_m = |\bar{h}(u)|$.

a) Montrer le *lemme d'itération sur les sous-mots* dont l'énoncé est :

Soit L un langage algébrique, alors il existe un entier $N > 0$ tel que si $u \in m(L)$ et $|u|_m \geq N$, on peut trouver cinq mots u_1, u_2, u_3, v et w satisfaisant les propriétés suivantes :

- 1) $u = u_1vu_2wu_3$;
- 2) $|vw|_m > 0$;
- 3) $|vu_2w|_m \leq N$;
- 4) $u_1v^k u_2w^k u_3 \in m(L)$ pour tout entier naturel k .

Indication. Un arbre de dérivation de u contient un sous-arbre de dérivation du sous-mot $\bar{h}(u)$: appliquer alors la méthode de démonstration du lemme d'itération à ce sous-arbre.

b) Montrer que le langage $L = \{a^m b^n c^p d^q \mid m = 0 \text{ ou } n = p = q\}$, où a, b, c et d sont distinctes deux à deux, n'est pas algébrique.

Indication. Montrer que $m(L)$ ne satisfait pas la conclusion du lemme d'itération sur les sous-mots.

c) Pour vérifier que le lemme d'itération sur les sous-mots est plus fin que le lemme d'itération, montrer que L satisfait la conclusion de ce dernier.

Indication. On pourra prendre $N = 1$.

Exercice 26. Lemme d'itération sur les facteurs.

Ce deuxième raffinement exploite encore l'idée précédente en utilisant les applications "facteur" $fact$ et $Fact$ (cf. l'exercice 1.17).

a) Montrer le *lemme d'itération sur les facteurs* dont l'énoncé est :

Soit L un langage algébrique, alors il existe un entier $N > 0$ tel que si $u = \alpha\beta\gamma \in L$ et $|\beta| \geq N$, on peut trouver cinq mots u_1, u_2, u_3, v et w satisfaisant les propriétés suivantes :

- 1) $u = u_1vu_2wu_3$;
- 2) v ou w est un facteur non vide de β ;
- 3) $u_1v^k u_2w^k u_3 \in L$ pour tout entier naturel k .

Indication. Reprendre la méthode de l'exercice précédent, avec les mots $\alpha\langle\beta\rangle\gamma$ et $\langle\beta\rangle$.

b) Montrer que le langage $L = \{a^m b^n c^p \mid m \neq n \text{ et } n \neq p\}$, où a, b et c sont distinctes deux à deux, n'est pas algébrique.

Indication. Montrer que L ne satisfait pas la conclusion du lemme d'itération sur les facteurs.

c) L'exemple qui va suivre permet de vérifier que l'itération sur les facteurs est plus fine que l'itération sur les sous-mots.

Soient $\mathcal{A} = a + b + c + d + e$ un alphabet constitué de 5 lettres distinctes deux à deux et

$$L = \{a^m b c b d^m b c b e^m \mid m \geq 0\} \\ + \{a^m c d^n b c b e^p \mid m, n, p \geq 0\} + \{a^m b c b d^n c e^p \mid m, n, p \geq 0\} \\ + \{u \in \mathcal{A}^* \mid |u|_b \geq 6\}$$

Montrer que L vérifie la conclusion du lemme d'itération sur les sous-mots mais qu'il n'est pas algébrique pour cela!

Indication. Montrer que L ne satisfait pas la conclusion du lemme d'itération sur les facteurs : pour tout entier $N > 0$ s'intéresser au mot $u = a^N b c b d^N b c b e^N$ factorisé avec $\alpha = \varepsilon, \beta = a^N$ et $\gamma = b c b d^N b c b e^N$.

Dérécursion à gauche.**Exercice 27. Des grammaires généralisées.**

Dans la définition d'une grammaire G , une règle globale pour $X \in \mathcal{V}$ est de la forme $X \longrightarrow \mathbf{l}(X)$ où $\mathbf{l}(X) \subseteq (\mathcal{V} + \mathcal{A})^*$ est supposé **fini**.

On dira que G est une *grammaire généralisée* si l'on suppose seulement que $\mathbf{l}(X)$ est algébrique pour chaque $X \in \mathcal{V}$.

a) Montrer que les langages engendrés par une grammaire généralisée sont algébriques. Appliquer ce résultat à un exemple simple.

b) On se propose de justifier la méthode de dérécursion à gauche évoquée dans la section 5.3.

Soit $A \in \mathcal{V}$. La règle globale de A peut s'écrire $A \xrightarrow{G} A\mathbf{p} + \mathbf{q}$ avec $\mathbf{p} \subseteq (\mathcal{A} + \mathcal{V})^*$ et $\mathbf{q} \subseteq (\mathcal{A} + \mathcal{V})^* - A(\mathcal{A} + \mathcal{V})^*$ (les éléments de \mathbf{q} ne commencent pas par la variable A). On peut considérer la grammaire généralisée $H = (\mathcal{V}, \mathcal{A}, \Sigma)$ dont les règles globales sont identiques à celles de G sauf pour la variable A dont la règle globale est maintenant $A \xrightarrow{H} \mathbf{q}\mathbf{p}^*$.

1) Montrer que pour toute $X \in \mathcal{V}$ on a $\mathcal{L}(H, X) = \mathcal{L}(G, X)$.

2) Construire une grammaire (non généralisée) équivalente à H (utiliser l'exercice 7).

Exercice 28. Une méthode matricielle.

Le second membre de la règle globale $X \longrightarrow \mathbf{l}(X)$ de toute $X \in \mathcal{V}$ peut s'écrire sous la forme

$$\mathbf{l}(X) = \sum_{Y \in \mathcal{V}} Y\mathbf{p}(YX) + \mathbf{q}(X)$$

où $\mathbf{p}(YX) \subseteq (\mathcal{A} + \mathcal{V})^*$ pour toute $Y \in \mathcal{V}$ et où $\mathbf{q}(X) \subseteq (\mathcal{A} + \mathcal{V})^* - \mathcal{V}(\mathcal{A} + \mathcal{V})^*$ est l'ensemble des éléments de $\mathbf{l}(X)$ qui ne commencent pas par une variable.

La dérécursion à gauche (cf. section 5.3 et exercice précédent) ne s'intéresse qu'à l'élimination des appels à gauche par une variable dans ses propres règles. La décomposition de \mathbf{l} ci-dessus met en évidence tous les appels à gauche et permet de traiter cette question de façon globale.

Pour ce faire, on considère l'alphabet $\mathcal{W} = \mathcal{V} \times \mathcal{V}$ (on notera $[XY]$ le couple $(X, Y) \in \mathcal{W}$), la substitution $\mathbf{m} : \mathcal{V} + \mathcal{W} \rightarrow \mathcal{P}((\mathcal{A} + \mathcal{V} + \mathcal{W})^*)$ définie par

$$\begin{aligned} - \mathbf{m}(X) &= \sum_{Y \in \mathcal{V}} \mathbf{q}(Y)[YX] + \mathbf{q}(X) \text{ pour toute } X \in \mathcal{V}, \\ - \mathbf{m}([XY]) &= \sum_{Z \in \mathcal{V}} \mathbf{p}(XZ)[ZY] + \mathbf{p}(XY) \text{ pour toute } [XY] \in \mathcal{W}, \end{aligned}$$

et enfin, $H = (\mathcal{V} + \mathcal{W}, \mathcal{A}, \Sigma)$ la grammaire dont les règles globales sont définies par \mathbf{m} .

a) Appliquer cette construction à la grammaire sur $\mathcal{A} = a + b$ et $\mathcal{V} = A_1 + A_2 + A_3$ dont les règles globales sont :

$$\begin{aligned} A_1 &\longrightarrow A_2A_3 + a \\ A_2 &\longrightarrow A_3A_1 + A_1b \\ A_3 &\longrightarrow A_1A_2 + A_2A_1 + b \end{aligned}$$

b) Montrer que l'on a $\mathcal{L}(H, X) = \mathcal{L}(G, X)$ pour toute $X \in \mathcal{V}$.

c) Montrer comment on peut utiliser cette transformation pour mettre une grammaire propre en forme normale de Greibach. Appliquer cette méthode à la grammaire ci-dessus (comparer le résultat obtenu ici à celui de la section 5. sur cette même grammaire, du point de vue du nombre de variables et de règles).

d) Montrer, en utilisant une mise en forme normale de Chomsky, que pour toute grammaire $G = (\mathcal{V}, \mathcal{A}, R)$ qui ne produit pas ε , il existe une grammaire équivalente (en un sens à préciser) dont chacune des règles a l'une des formes suivantes :

$$X \longrightarrow xYZ \quad X \longrightarrow xY \quad X \longrightarrow x$$

où $x \in \mathcal{A}$ et X, Y et $Z \in \mathcal{V}$.

